

Toolbox for the Modeling and Analysis of Thermodynamic Systems Workshop

Ohio Aerospace Institute

April 15, 2015

Agenda

8:30-9 **Registration**

9-12 Welcome/Introduction

Expectations

Propulsion System Simulation Using the Toolbox for the Modeling and
Analysis of Thermodynamic Systems (T-MATS)

T-MATS Demo

Development of a twin-spool turbofan engine simulation using the Toolbox
for Modeling and Analysis of Thermodynamic Systems (T-MATS)

A Process for the Creation of T-MATS Propulsion System Models from NPSS Data

Cantera Integration with the Toolbox for Modeling and Analysis of
Thermodynamic Systems (T-MATS)

T-MATS Simulation of Engine Performance During Ice Particle Ingestion

12-1 lunch

1-3 **External Presentations and NASA Project Presentations**

Model Extraction by System Identification and Advanced Control Design in T-MATS

Using T-MATS for Supporting the Design of Bleed System Controls

Advanced Propulsion Control Technologies

Model-Based Engine Control using T-MATS

3-4 Discussion/Future Work

Wrap-up

Sanjay Garg/Jonathan Litt

Jonathan Litt

Jeff Chapman

Jeff Chapman

Alicia Zinnecker

Jeff Chapman

Tom Lavelle

Jeff Chapman

Hanz Richter, Cleveland State University

Umer Khan, Honeywell Canada

Dennis Culley

Jeff Csank

all

all

T-MATS

- T-MATS is a Simulink Blockset for modeling thermodynamic systems
- It was primarily designed to model turbomachinery, but it is not limited to that
- Because it is in Simulink, control design, dynamic analysis, and testing are simplified
- T-MATS is a work in progress
- It is a powerful tool, but it can be improved
- A Goal of this Workshop is to see what improvements would be valuable to the user community

T-MATS Development

- T-MATS is an Open Source Software package created at NASA Glenn
- T-MATS was created to solve specific problems
- Internal T-MATS development is supported by projects
- Internal T-MATS development does not generally occur for its own sake
- However, Open Source software encourages collaborative development, and user-defined blocks can be posted on our github site
- A desired Workshop Outcome is to define a prioritized list of desired improvements
- This list can be used to advocate for additional development

Expectations

- Attendees will learn about T-MATS and its current features
- Presenters will describe various T-MATS applications, current and planned
- Model developers will talk about their experience with T-MATS:
 - What features are good and why?
 - Can they be improved?
 - What is missing?
- The group will discuss new feature development options. Can we reach a consensus?
- The group will prioritize these potential new features



Propulsion System Simulation Using the Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS)

Jeffryes W. Chapman

Thomas M. Lavelle

Ryan D. May

Jonathan S. Litt

Ten-Huei (OA) Guo

T-MATS Workshop

Cleveland, OH

April 15, 2015



Acknowledgments

Funding for this work was provided by NASA Aviation Safety Program's Vehicle Systems Safety Technology Project (VSST)



Outline

- Background
- T-MATS Description
- Framework
- Block Sets
- Examples
- Summary
- References and Download information



Background, Modeling Goals

- Requirements for transient gas turbine simulation for academia, research, or industry.
 - Flexible plant model
 - Sets of components that may be used to create custom turbo-machinery performance models.
 - Ability to leverage legacy model design and codes
 - Numerical solvers for system convergence
 - Dynamic operation for transient simulation
 - Ability to easily create a dynamic model from a steady state model
 - Faster than real time operation
 - Easy integration with common design tools
 - Seamless integration with or built in MATLAB®/Simulink®.
 - Parameterized and easily modifiable.
 - Ability to collaborate with international workforce
 - Non-proprietary, free of export restrictions, and open source.

No single software package meets all of these requirements

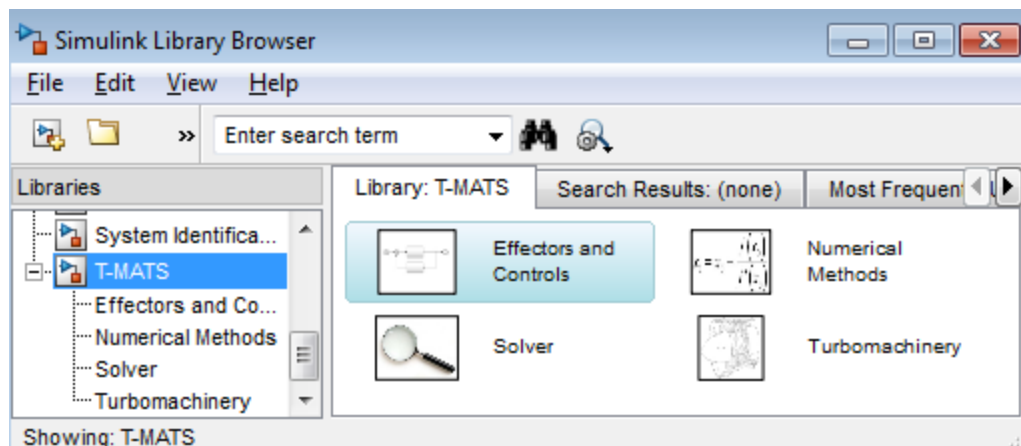


T-MATS Description

- **Toolbox for the Modeling and Analysis of Thermodynamic systems, T-MATS**
 - Modular thermodynamic modeling framework
 - Designed for easy creation of custom Component Level Models (CLM)
 - Built in MATLAB/Simulink
- **Package highlights**
 - General thermodynamic simulation design framework
 - Variable input system solvers
 - Advanced turbo-machinery block sets
 - Control system block sets
- **Development being led by NASA Glenn Research Center**
 - Non-proprietary, free of export restrictions, and open source
 - Open collaboration environment

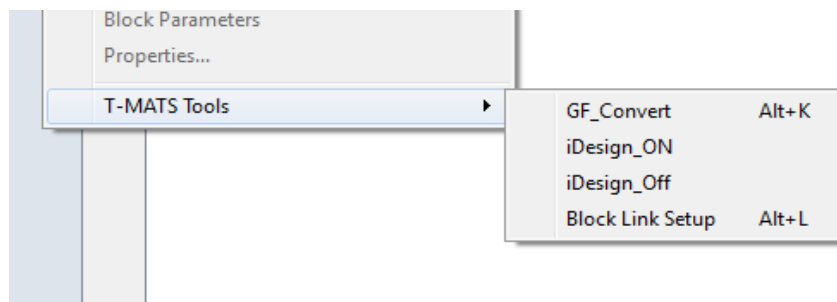
T-MATS Framework

- T-MATS is a plug-in for a MATLAB/Simulink platform
 - additional blocks in the Simulink Library Browser:



Added Simulink
Thermodynamic
modeling and numerical
solving functionality

- additional diagram tools for model development in Simulink:

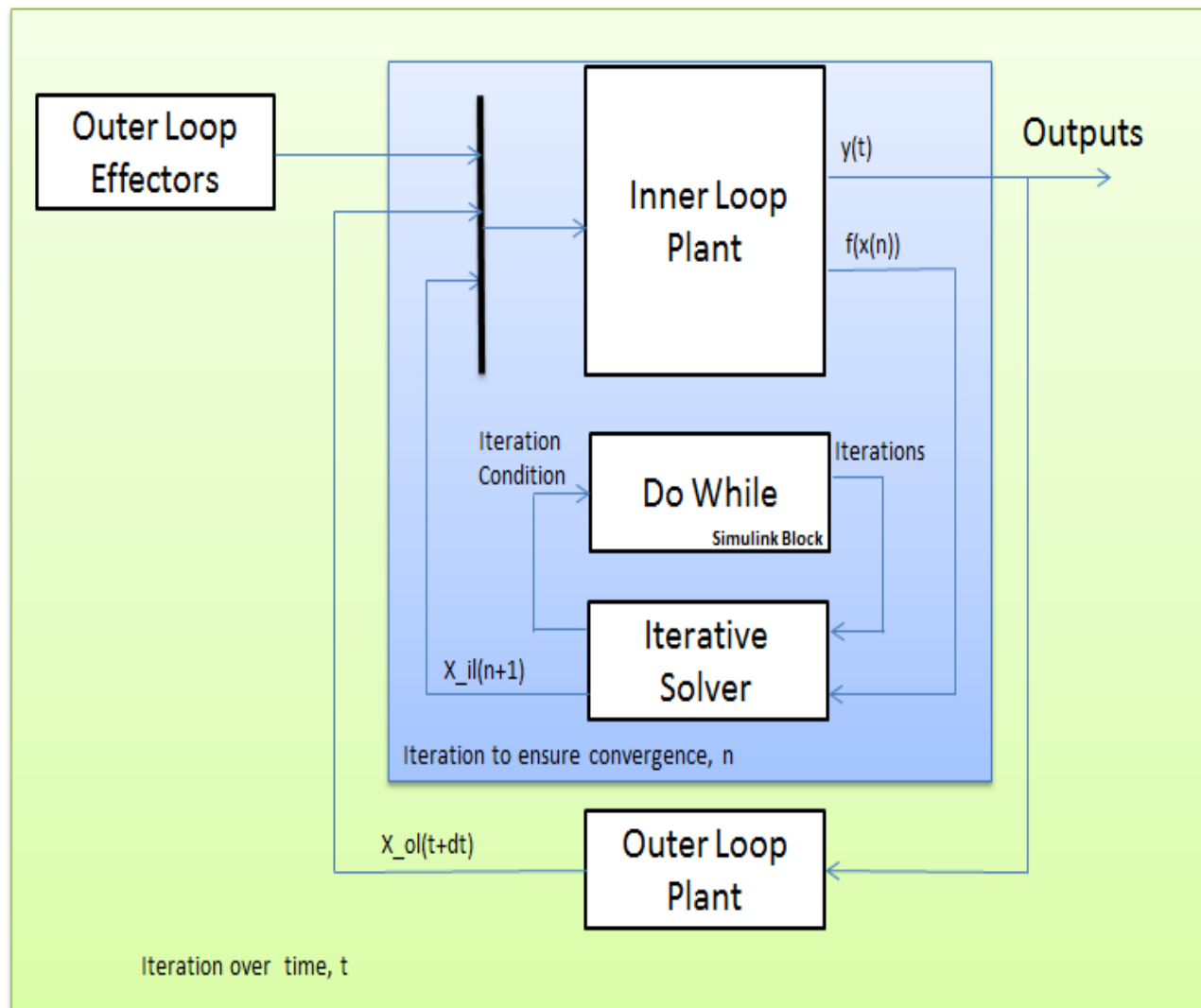


Faster and easier
model creation

T-MATS Framework

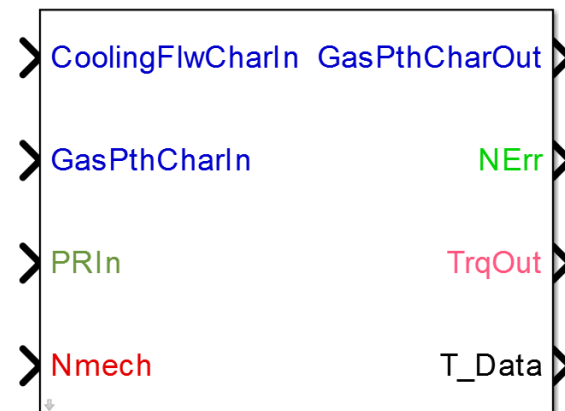
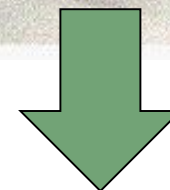
Dynamic Simulation Example:

- Multi-loop structure
 - The “outer” loop (green) iterates in the time domain
 - Not required for steady-state models
 - The “inner” loop (blue) solves for plant convergence during each time step



Blocks: Turbo-machinery

- T-MATS contains component blocks necessary for creation of turbo-machinery systems
 - Modeling theory based on common industry practices
 - Energy balance modeling approach
 - Compressor models utilize R-line compressor maps
 - Turbine models utilize Pressure Ratio turbine maps
 - Single fuel assumption (non-Cantera version)
 - Blocks types; compressor, turbine, nozzle, flow splitter, and valves among others.
 - Color Coding for easy setup
 - Built with S-functions, utilizing compiled C code/ MEX functions

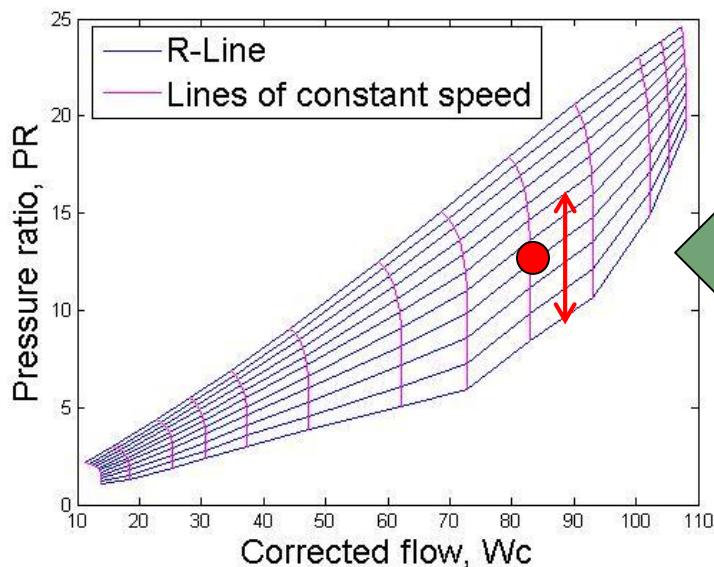
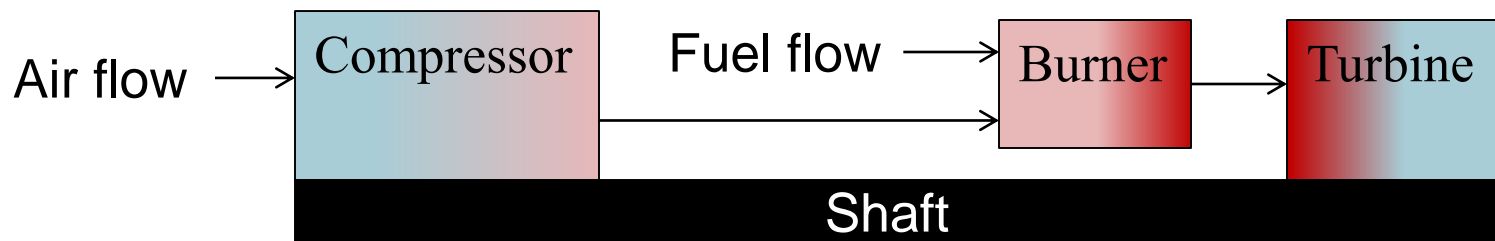


Turbine

Blocks: Numerical Solver

- Why is an external solver necessary?

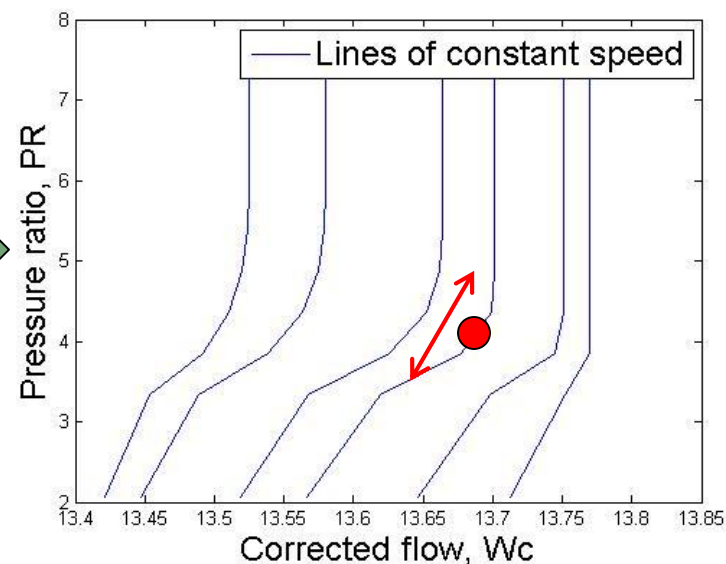
- Many thermodynamic simulations contain variables that are system dependent.
- In Gas turbine models air flow through the engine is dependent on system architecture.



Components must agree on W for the system

Convergence

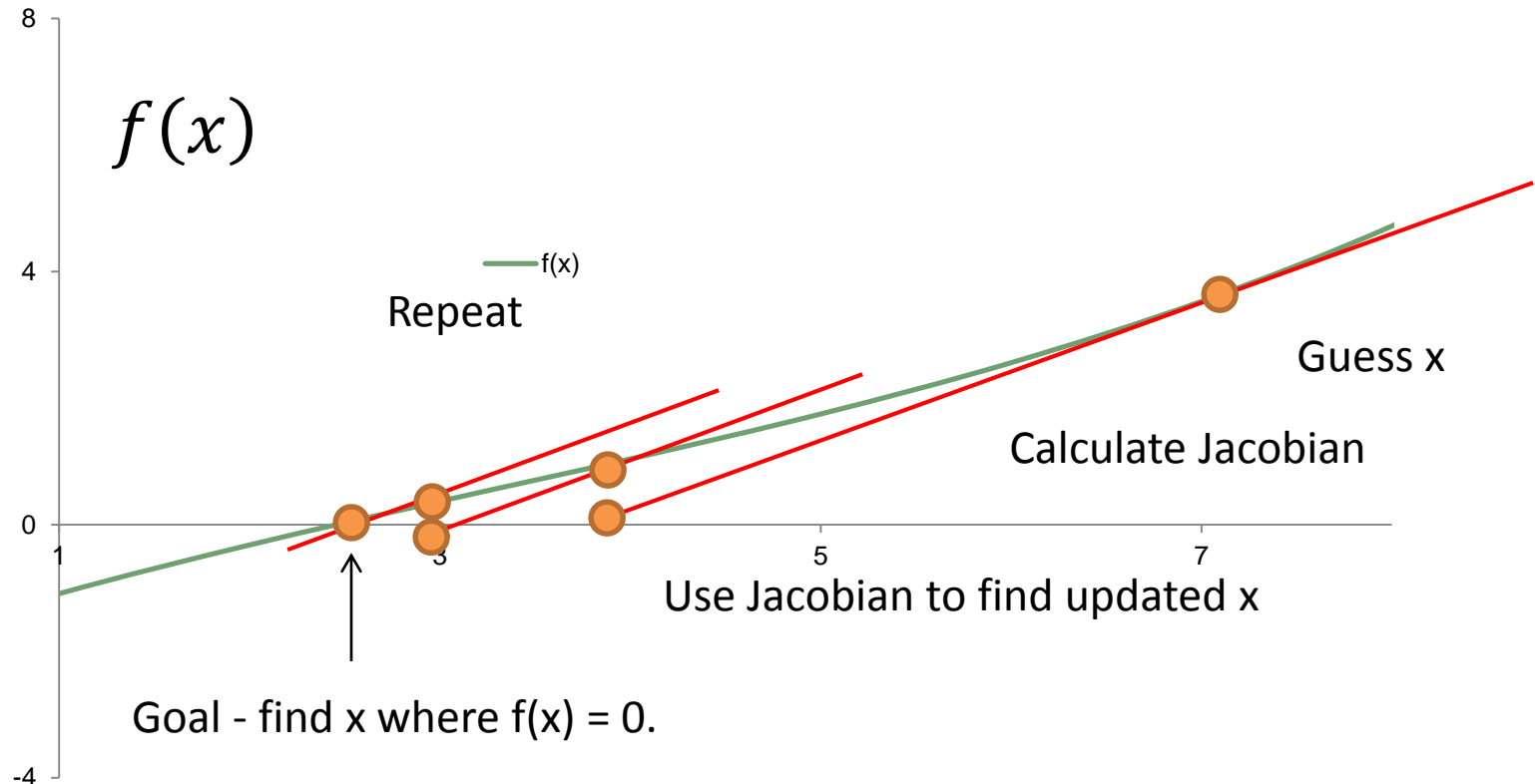
Effectors:
Shaft speed
Pressure
Temperature



Blocks: Numerical Solver

- T-MATS solvers utilize the Newton Raphson method

$$x(k+1) = x(k) - \frac{f(x(k))}{\nabla f(x(k))} \quad \text{where,} \quad \nabla f(x(k)) = \text{Jacobian}$$

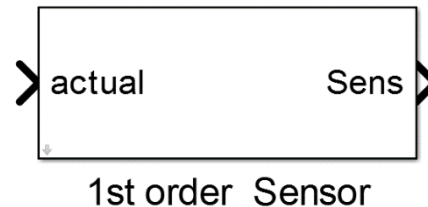




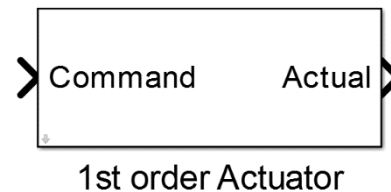
Blocks: Controls

- T-MATS contains component blocks designed for fast control system creation

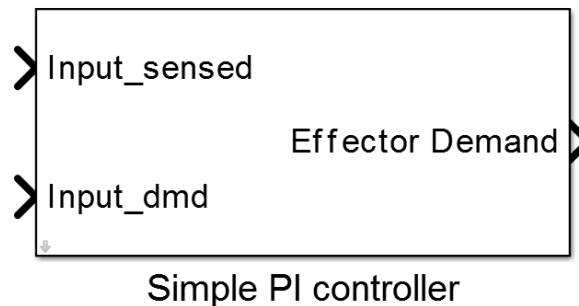
- Sensors:



- Actuators:

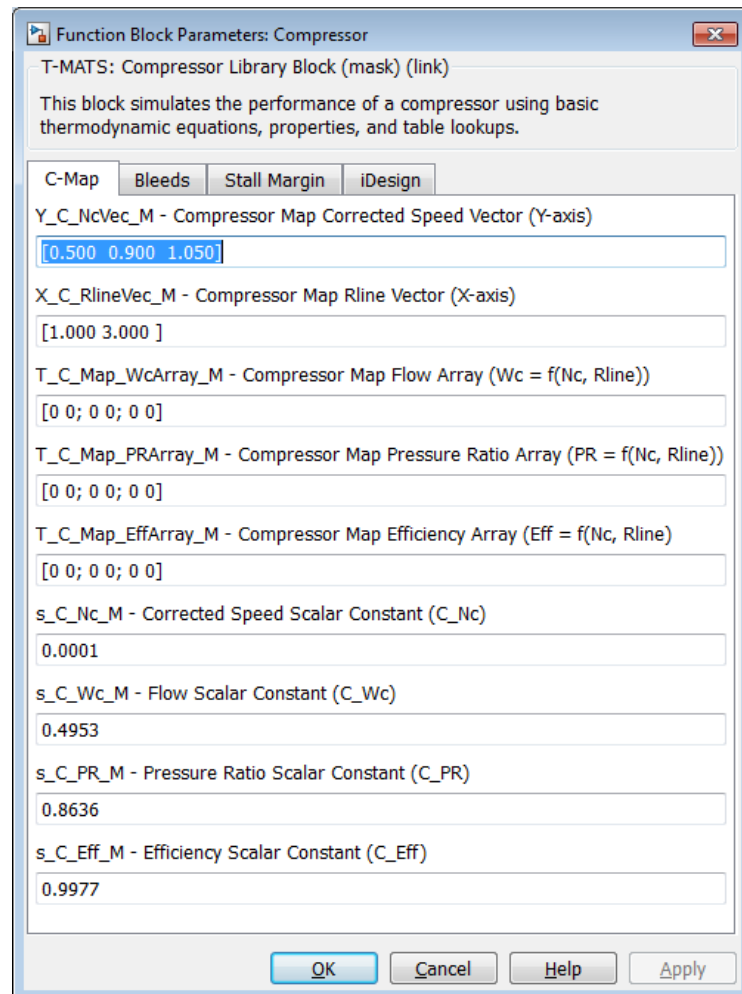


- PI controllers:



Blocks: Settings

- The T-MATS Simulation System is a highly tunable and flexible framework for Thermodynamic modeling.
 - T-MATS block Function Block Parameters
 - fast table and variable updates
 - Open source code
 - flexibility in component composition, as equations can be updated to meet system design
 - MATLAB/Simulink development environment
 - user-friendly, powerful, and versatile operation platform for model design



Function Block Parameters: Compressor

T-MATS: Compressor Library Block (mask) (link)

This block simulates the performance of a compressor using basic thermodynamic equations, properties, and table lookups.

C-Map Bleeds Stall Margin iDesign

Y_C_NcVec_M - Compressor Map Corrected Speed Vector (Y-axis)

[0.500 0.900 1.050]

X_C_RlineVec_M - Compressor Map Rline Vector (X-axis)

[1.000 3.000]

T_C_Map_WcArray_M - Compressor Map Flow Array ($Wc = f(Nc, Rline)$)

[0 0; 0 0; 0 0]

T_C_Map_PRArray_M - Compressor Map Pressure Ratio Array ($PR = f(Nc, Rline)$)

[0 0; 0 0; 0 0]

T_C_Map_EffArray_M - Compressor Map Efficiency Array ($Eff = f(Nc, Rline)$)

[0 0; 0 0; 0 0]

s_C_Nc_M - Corrected Speed Scalar Constant (C_{Nc})

0.0001

s_C_Wc_M - Flow Scalar Constant (C_{Wc})

0.4953

s_C_PR_M - Pressure Ratio Scalar Constant (C_{PR})

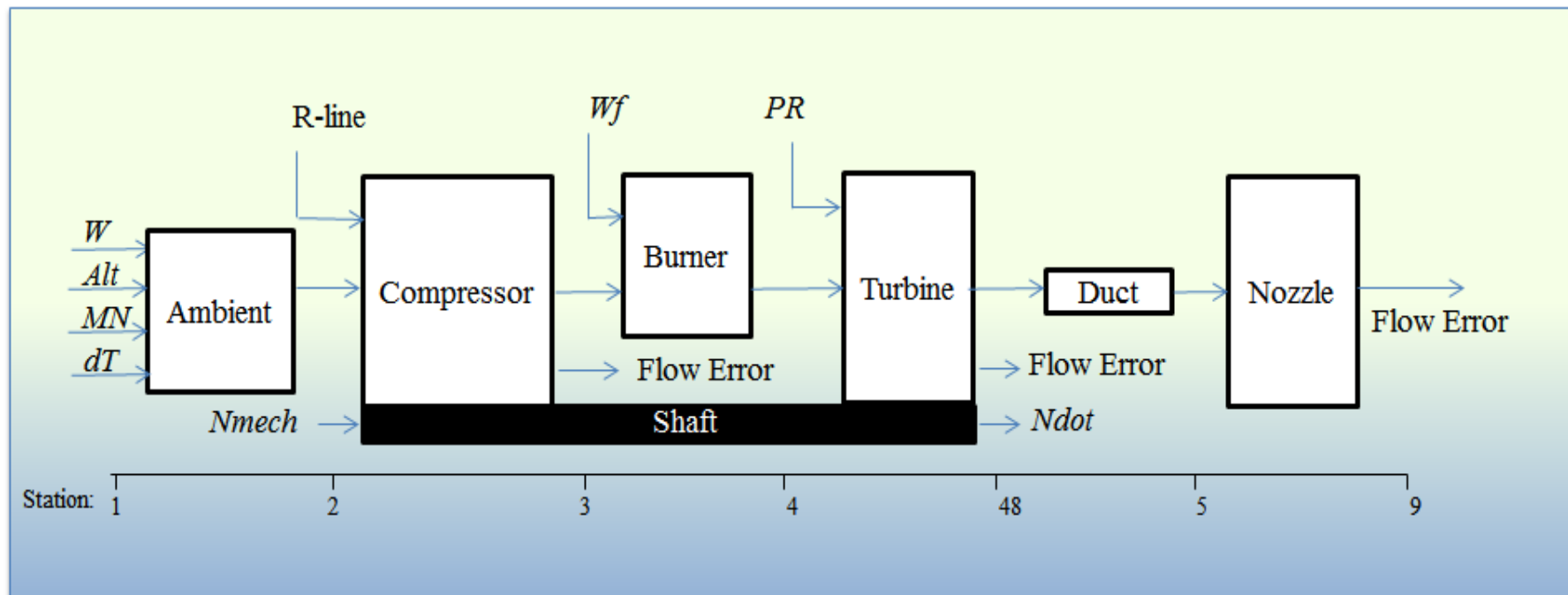
0.8636

s_C_Eff_M - Efficiency Scalar Constant (C_{Eff})

0.9977

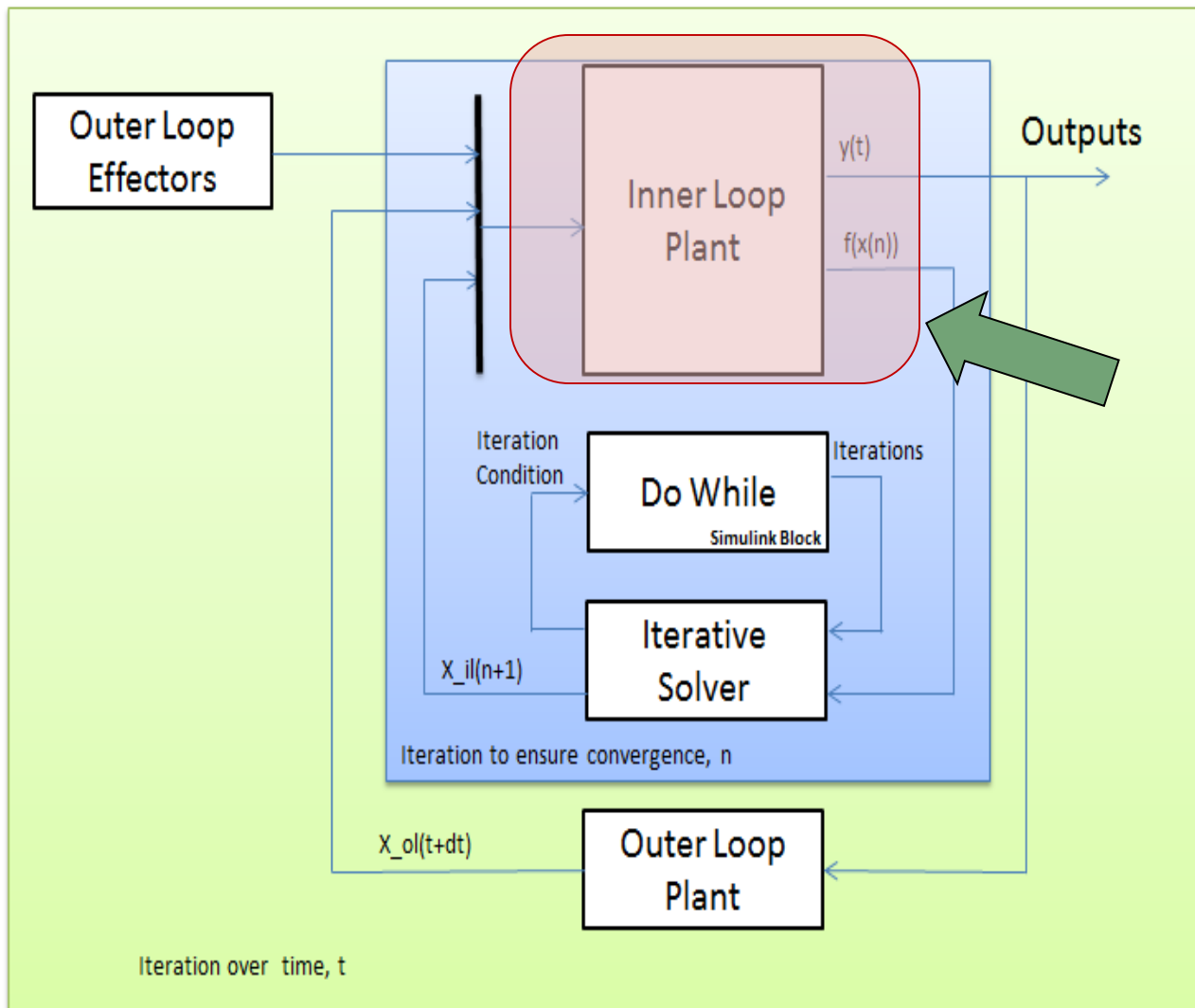
OK Cancel Help Apply

Dynamic Gas Turbine Example: Objective System

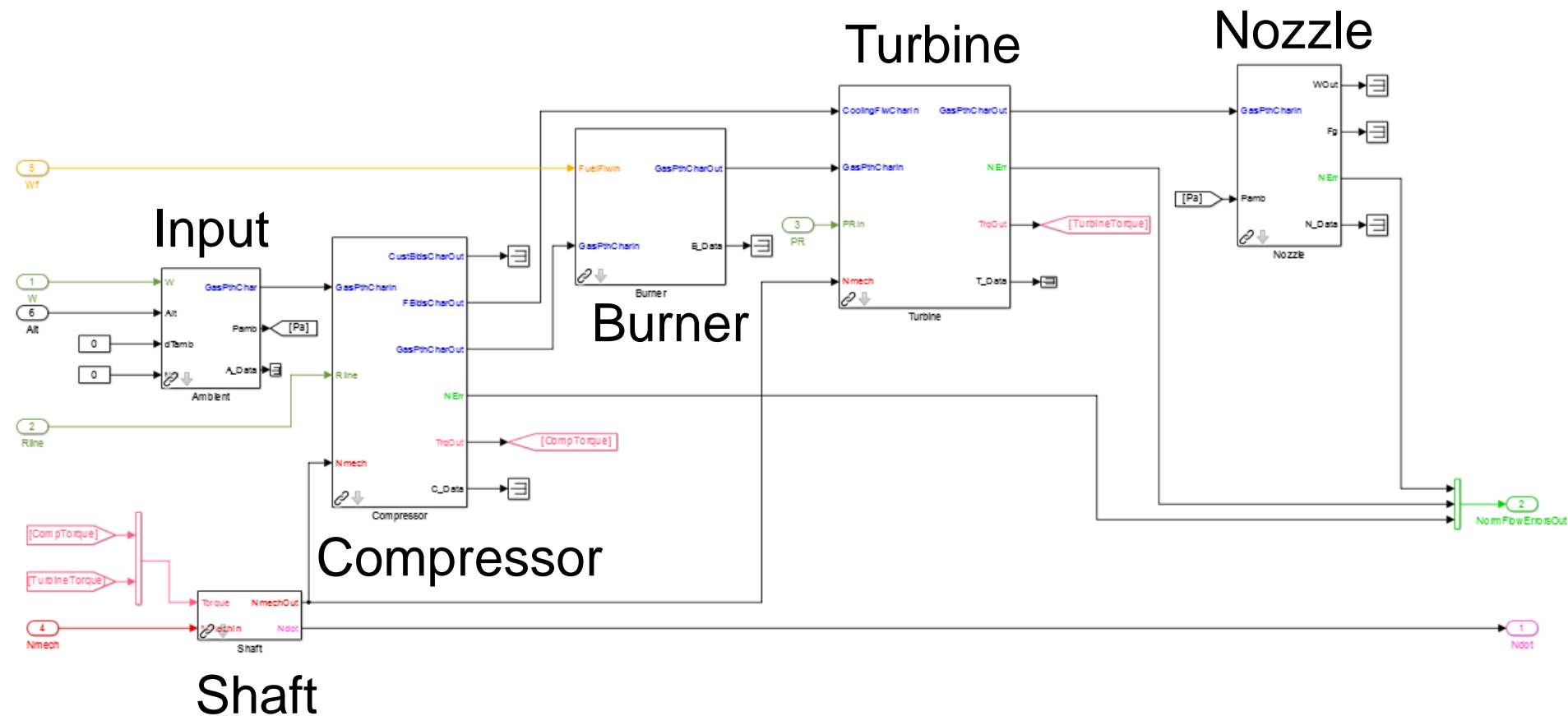


Simple Turbojet

Dynamic Gas Turbine Example: Creating the Inner Loop

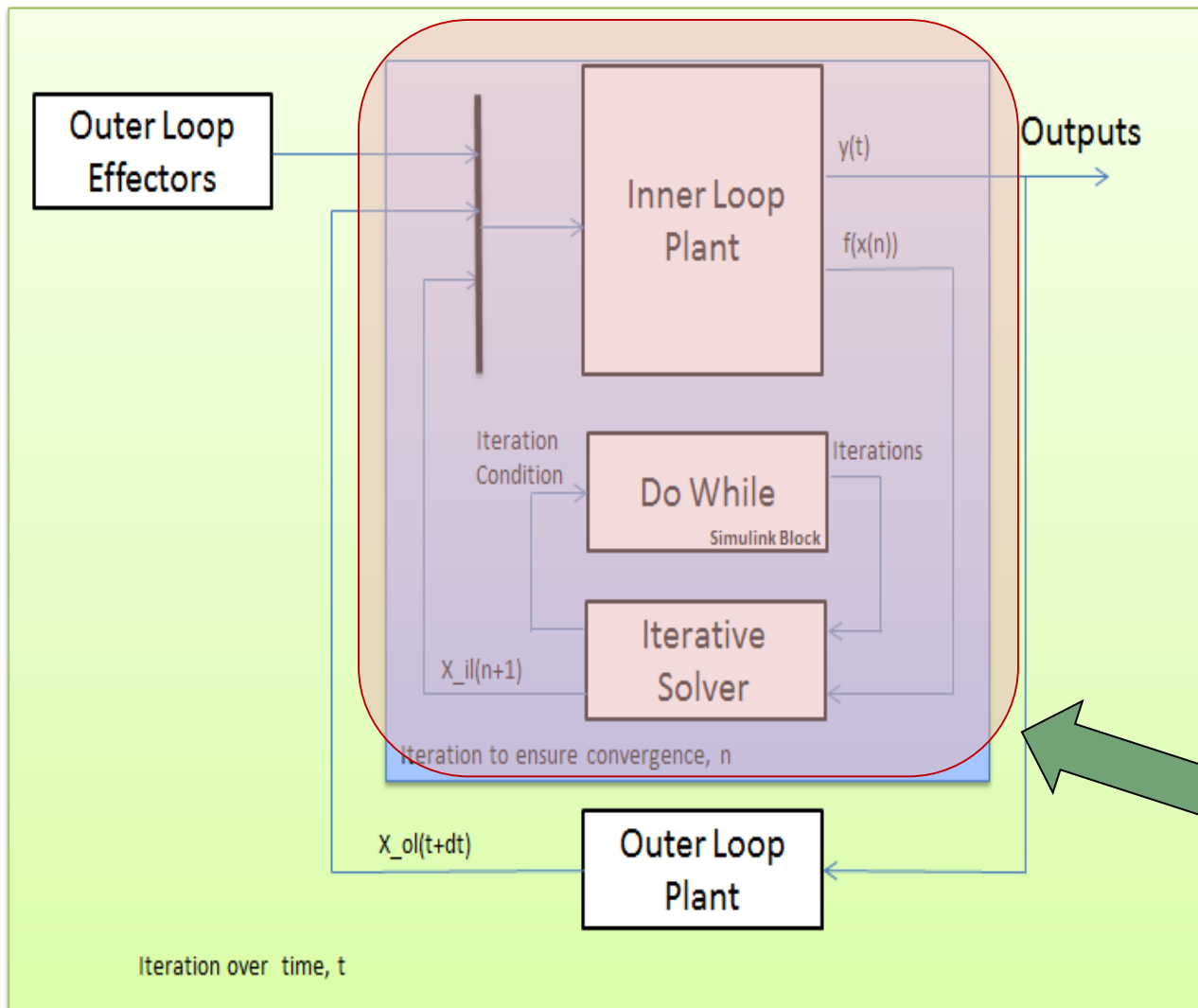


Dynamic Gas Turbine Example: Inner Loop Plant



Turbojet plant model architecture made simple by T-MATS vectored I/O and block labeling

Dynamic Gas Turbine Example: Creating the Solver



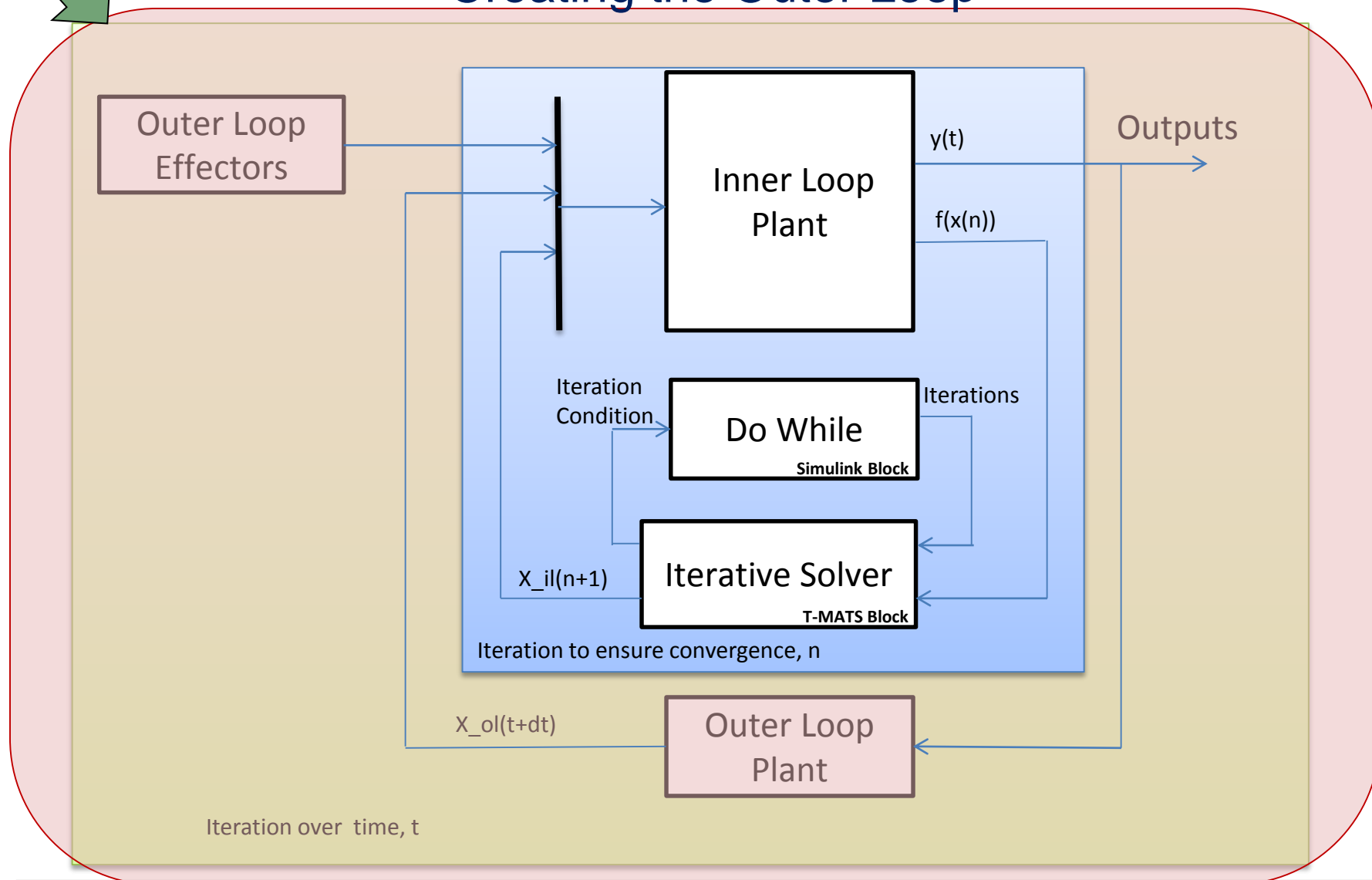
Simulink While Iterator block



Iterative Solver

www.nasa.gov 17

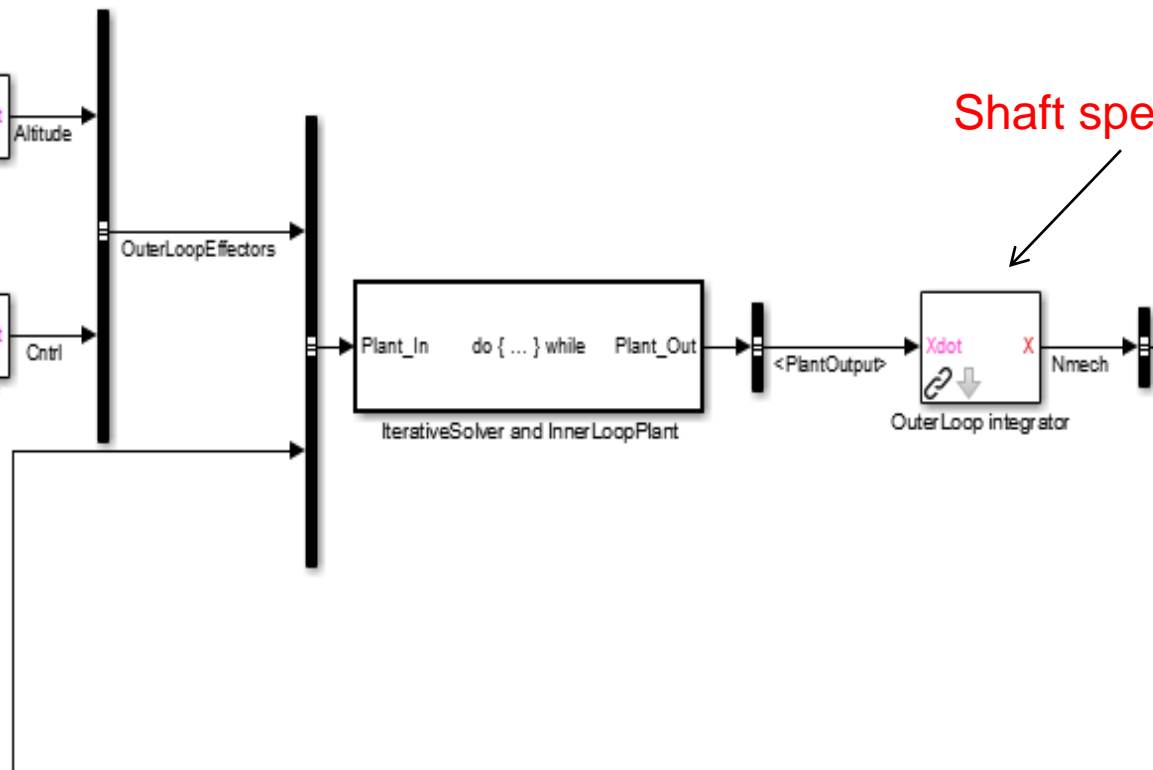
Dynamic Gas Turbine Example: Creating the Outer Loop



Dynamic Gas Turbine Example: Outer Loop Plant

Environmental
conditions

Simple Control
System

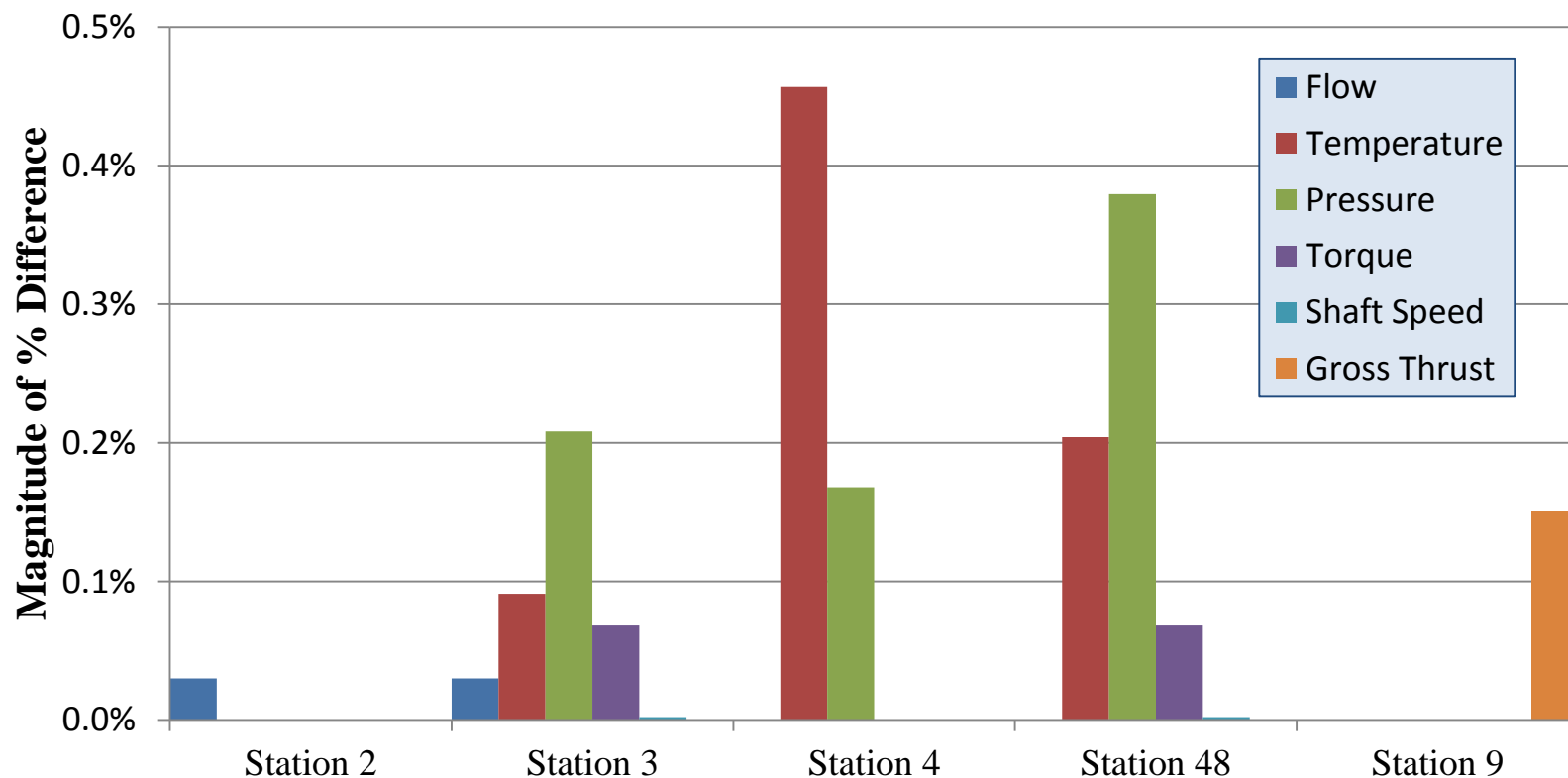


Shaft integrator and other Outer Loop effectors added to create full system simulation

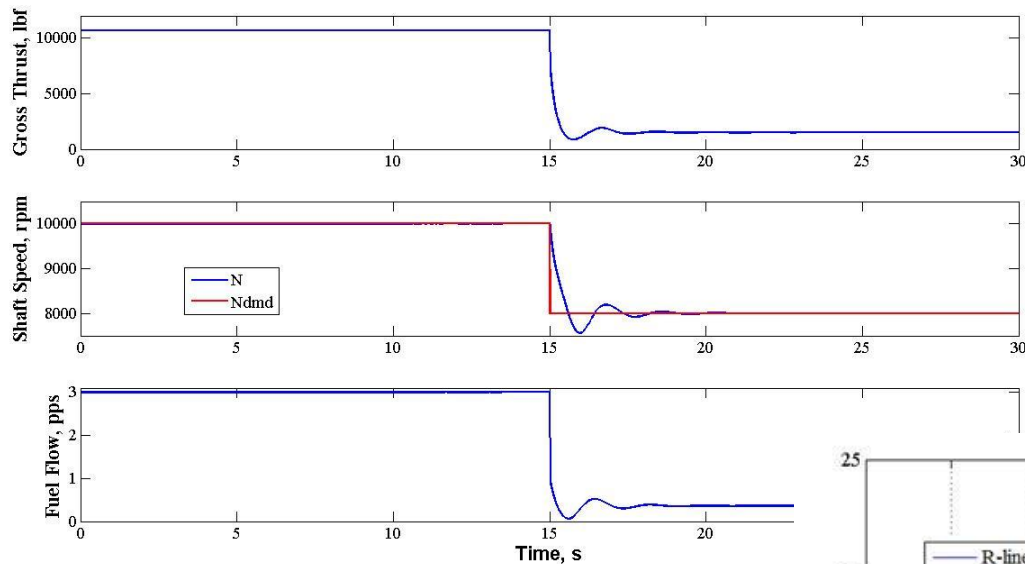


Example Model Match

- Data generated from the example T-MATS turbojet compared to a steady state “truth” model developed in NPSS.
 - All difference values less than 0.5%



Example Dynamic Operation

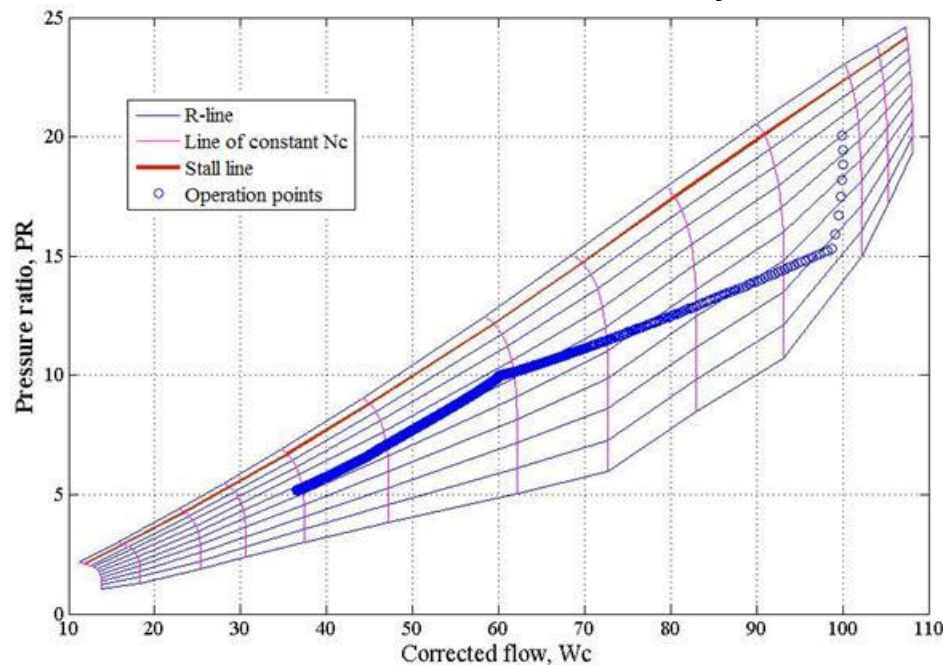


- Simulation of real time system.

- Engine maneuvers
- Control system performance
- Sensor delays

- Dynamic Events

- Engine response times
- Stall margin modeling capability
- Simulation of fault transients





Summary

- T-MATS offers a powerful and user-friendly simulation environment for propulsion system modeling
 - Thermodynamic system modeling framework
 - Automated system “convergence”
 - Advanced turbo-machinery modeling capability
 - Fast controller creation block set
 - Capable of running faster than real time
 - Plug-in for Simulink



References and Download Information

- Download information may be found at:
<https://github.com/nasa/T-MATS/releases/>

- References:

1. Chapman, J.W., Lavelle, T.M., May, R.D., Litt, J.S., and Guo, T.M., "Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS) User's Guide," NASA/TM-2014-216638, January 2014.
2. Chapman, J.W., Lavelle, T.M., May, R.D., Litt, J.S., Guo, T-H., "Propulsion System Simulation Using the Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS)," 2014 AIAA Joint Propulsion Conference, Cleveland, OH, Jul 28-30, 2014.
3. Lavelle, T.M., Chapman, J.W., May, R.D., Litt, J.S., and Guo, T.H., "Cantera Integration with the Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS)," 2014 AIAA Joint Propulsion Conference, Cleveland, OH, Jul 28-30, 2014.
4. Chapman, J.W., Lavelle, T.M., Litt, J.S., Guo, T-H., "A Process for the Creation of T-MATS Propulsion System Models from NPSS Data," 2014 AIAA Joint Propulsion Conference, Cleveland, OH, Jul 28-30, 2014.
5. Zinnecker, A.M., Chapman, J.W., Lavelle, T.M., and Litt, J.S., "Development of a twin-spool turbofan engine simulation using the Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS)," 2014 AIAA Joint Propulsion Conference, Cleveland, OH, Jul 28-30, 2014.



Development of a twin-spool turbofan engine simulation using the Toolbox for Modeling and Analysis of Thermodynamic Systems (T-MATS)

Alicia M. Zinnecker
N&R Engineering

Jeffreys W. Chapman
Vantage Partners, LLC

Thomas M. Lavelle
Jonathan S. Litt
NASA Glenn Research Center

Intelligent Control and Autonomy Branch
NASA Glenn Research Center

T-MATS Workshop
April 15, 2015



Outline

- 1 Introduction
- 2 Engine plant models
 - C-MAPSS engine plant model
 - T-MATS engine plant model
 - Comparison of models
- 3 Evaluating the engine models
 - Simulation setup
 - Comparison of simulation results: Steady-state
 - Comparison of simulation results: Dynamic
- 4 Summary and conclusion

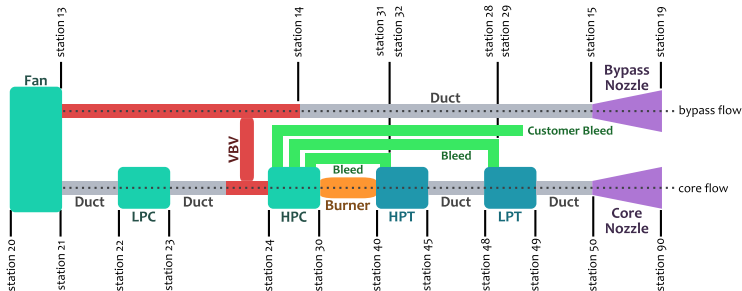


Introduction

- What is T-MATS?
 - Toolbox for **M**odeling and **A**nalysis of **T**hermodynamic **S**ystems
 - A library for MATLAB/Simulink® (The MathWorks, Inc.) containing **customizable thermodynamic element models**
- To demonstrate the modeling an engine using T-MATS, the C-MAPSS engine was recreated using this tool
 - **Commercial Modular Aero-Propulsion System Simulation**, a 0-d simulation of a 90,000 lb_f thrust twin-spool turbofan engine
 - Models considered to 'match' when the following were **within 1%** between the models:
 - Flow characteristics (flow rate, enthalpy, temperature, pressure, and fuel-to-air ratio) at exit of each component
 - R-lines, pressure ratios for compressors, turbines
 - Fan and core shaft speeds
 - Thrust produced by bypass and core nozzles
 - Matching of the models was verified at **component** and **system** levels, through **steady-state** and **dynamic** simulation



C-MAPSS engine plant model



- Main components:

- Three compressors (fan, low pressure compressor, high pressure compressor)
- Two turbines (low pressure and high pressure turbines)
- Burner
- Two nozzles (bypass and core flow path)
- Five ducts located throughout flow paths



C-MAPSS engine plant model

- **Engine and controller model** implemented in MATLAB/Simulink®
- Simulation defined by **input profiles** supplied to model
 - Environmental inputs: altitude, Mach number, ambient temperature
 - Health inputs for engine components
 - Variable geometry: variable stator vanes (VSV), variable bleed valve (VBV)
 - Fuel flow input (open-loop) or throttle command input (closed-loop)
- **Compressor, turbine maps** characterize five main engine components
 - Maps **relate** (corrected) flow rate (W_c), efficiency (η), pressure ratio (PR), and (corrected) shaft speed (NR)
 - Sizing of components using **scalers**

$$W_c^{un} = s_{W_c}^C W_c^{map}$$

$$NR^{map} = s_{NR}^C NR^{un}$$

$$\eta^{un} = s_{\eta}^C \eta^{map}$$

$$\begin{cases} PR^{un} = s_{PR}^C (PR^{map} - 1) + 1 & \text{for compressors} \\ PR^{map} = s_{PR}^C (PR^{un} - 1) + 1 & \text{for turbines} \end{cases}$$

- **Unique operating points** determined by R-line (compressors) or PR (turbines)



C-MAPSS engine plant model

- **Iterative solver** used to **maintain balanced flow** throughout simulation
 - Reduce **five** flow errors to below specified tolerance at each time step
 - 1 Error between flow exiting fan and flow exiting each branch of VBV
 - 2 Error between flow entering and exiting high-pressure compressor
 - 3 Error between flow entering and exiting high-pressure turbine
 - 4 Error between flow entering and exiting low-pressure turbine
 - 5 Error between flow entering and exiting core nozzle
 - Adjust **R-lines, pressure ratios** to each compressor, turbine to define the operating point at which flow is balanced



T-MATS engine plant model

- The T-MATS engine model is constructed by placing blocks from the T-MATS library **for each component** of the C-MAPSS engine model
 - Ducts, valve, and splitter blocks needed in addition to compressor, turbine, burner, and nozzle blocks
 - Configure blocks by specifying **maps** and **physical attributes** defined for C-MAPSS
- To achieve matching with the C-MAPSS engine, several **modeling assumptions** needed to be addressed
 - Done through modifications to **either** the C-MAPSS maps, scalars, etc. or to the T-MATS code
 - **Rescale compressor and turbine maps** to match scaling relationship assumed in T-MATS code

$$\begin{aligned}
 (\cdot)^{un} &= s_{(\cdot)}^T (\cdot)^{map} && \text{for values read from maps} \\
 (\cdot)^{map} &= s_{(\cdot)}^T (\cdot)^{un} && \text{for values used for interpolation}
 \end{aligned}$$

- **Recalculate fractional bleed** to high-pressure turbine, assume no customer bleed

$$\begin{aligned}
 W_{31,32} &= f_{bld}^{31,32} (W_{24} - W_{28} - W_{29} - W_{cust}) \\
 &= \bar{f}_{bld}^{31,32} W_{24} - f_{bld}^{31,32} W_{cust}
 \end{aligned}$$



T-MATS engine plant model

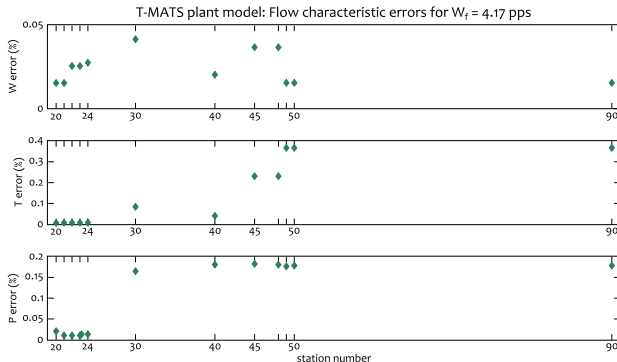
- Recalculation of component configuration data addressed **most** of the mismatch between models
- **Code-level modification** of T-MATS turbine block required so flow error calculations matched data in maps
 - C-MAPSS assumes maps contain **inflow only** ($W_{40,48}$); T-MATS assumes **inlet bleed flow** ($W_{31,28}$) is **also** captured
 - Change **one line** at end of code, replacing $W_{s1} = W_{40,48} + W_{31,28}$ with $W_{40,48}$

$$\begin{aligned}e_{HPT,LPT}^C &= W_{48,50} - W_{31,28} - W_{32,29} - W_{40,48} \\ &= W_{40,48}^{map} - W_{40,48}\end{aligned}$$

$$\begin{aligned}e_{HPT,LPT}^T &= (W_{40,48} + W_{31,28}) + W_{32,29} - W_{40,48}^{map} \\ &= W_{40,48} + W_{31,28} - W_{40,48}^{map}\end{aligned}$$



Comparing the models



- Comparison of **flow rate**, **temperature**, **pressure** at each station, for input fuel flows $W_f = 1.67$ pps, 3.33 pps, 4.17 pps (shown here), 6.95 pps
- Models match well: flow characteristics at each station in the T-MATS model **within 1%** of the C-MAPSS model
- **No iterative solver** has been implemented in this model, flow not guaranteed to be balanced



Simulating the engine models

- Model needs mechanism to **ensure mass conservation** throughout simulation
 - Flow is 'balanced' when **input** and **output** flow of each component are **within a tolerance** of each other
 - R-line, PR determines flow in compressors, turbines (from maps)
 - Iterative solver adjusts these to reduce flow errors
- T-MATS contains **two** solvers
 - **Steady-state solver**: balances the flow for a given *constant input*
 - **Dynamic solver**: drives flow errors to 0 *at each time step*
- Solver configuration specifies **when flow is considered 'balanced'** and **limitations on internal calculations**
 - Most influential in how well models matched was solver **termination condition**
 - **Steady-state** solver set to stop once errors **within 1%**
 - Investigations suggested modifying **dynamic solver** condition to **0.01%** to improve matching (without impacting performance)



Simulating the engine models: Simulation Setup

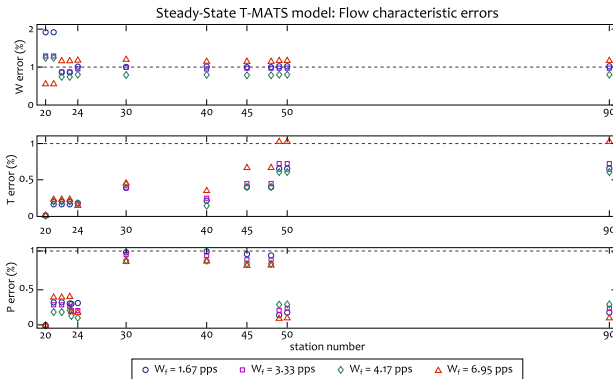
- Both engine models simulated at **sea level static conditions** with **eight fuel flow profiles**, covering large range of operating conditions

Simulation	Initial fuel flow (pps)	Final fuel flow (pps)
SS1	1.67	1.67
SS2	3.33	3.33
SS3	4.17	4.17
SS4	6.95	6.95
D12	1.67	3.33
D13	1.67	4.17
D24	3.33	6.95
D34	4.17	6.95

- Comparison of the models was done through:
 - Calculating the **difference in flow characteristics** at each station, as a percent of C-MAPSS simulation results
 - Comparing the **iterative solver outputs**
 - Verifying that **flow errors** were reduced below the specified tolerance



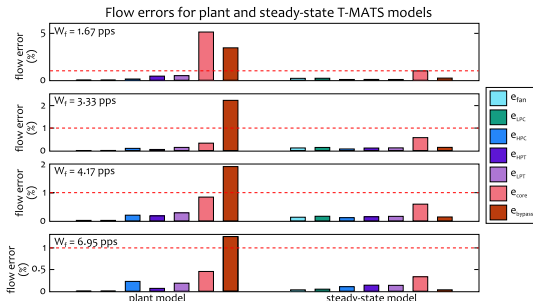
Simulating the engine models: Steady-state solver



- **Steady-state simulation results** for flow rate (W), temperature (T), and pressure (P) at all stations in core flow path
- Temperature and pressure generally $< 1\%$, flow rate around 1%
- Larger differences than when T-MATS engine plant model was compared because **flow is unbalanced without an iterative solver**



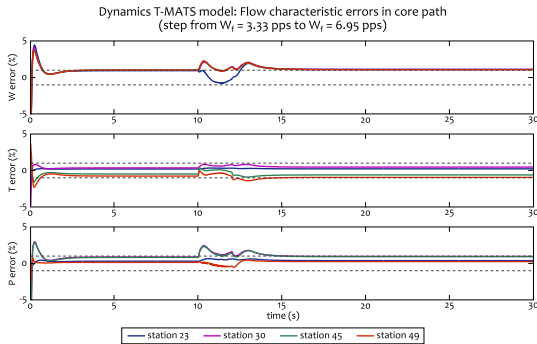
Simulating the engine models: Steady-state solver



- Flow errors in T-MATS engine plant model **much larger** than in steady-state model
- Bypass nozzle flow error, in particular, exceeds solver error tolerance
 - Iterative solver drives error below tolerance, converging model to a steady-state **slightly different** from the C-MAPSS steady-state
 - Large flow error is related to **larger iterative solver dimension** (5 versus 7 flow errors) and related **differences in flow error calculation**



Simulating the engine models: Dynamic solver

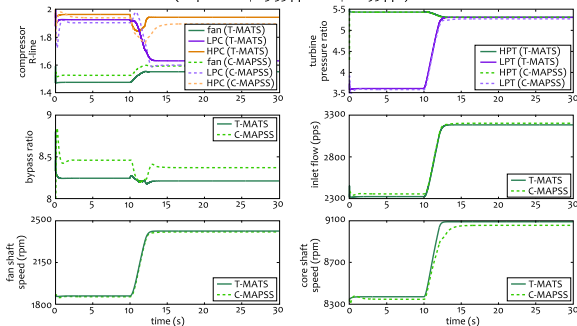


- Shown here are **time plots of W, T, P** at exits of low- and high-pressure compressors and turbines for simulation with fuel flow increasing from $W_f = 3.33$ pps to $W_f = 6.95$ pps
- The T-MATS model is **generally around 1%** of the C-MAPSS model
 - Largest mismatch occurs **shortly after flow transition**, related to implementation of variable bleed valve input
 - Start-up transient causes small differences early in simulation



Simulating the engine models: Dynamic solver

Dynamic T-MATS model: Iterative solver results and shaft speeds
(step from $W_f = 3.33$ pps to $W_f = 6.95$ pps)

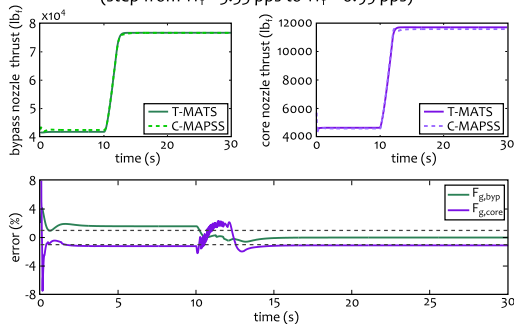


- Comparison of solver outputs shows **good matching** of **turbine pressure ratios** and **shaft speeds**
- Mismatch in compressor R-lines, inlet flow, and bypass ratio (BPR), also observed in steady-state T-MATS model, due to **different flow error calculations** in the two models



Simulating the engine models: Dynamic solver

Dynamic T-MATS model: Thrust production
(step from $W_f = 3.33$ pps to $W_f = 6.95$ pps)



- Additional outputs of interest for comparing models are **thrusts produced through each nozzle**
- Thrust is **function of flow rate** through nozzle, so the 1 – 2% difference in W_{19} and W_{90} produce a similar difference in thrust produced by core and bypass nozzles, respectively



Summary and conclusion

- An **accurate open-loop replica** of the C-MAPSS engine model has been constructed using the T-MATS Simulink library
 - Required understanding of **several model assumptions** made in C-MAPSS and T-MATS about **component maps, scalars, and flow error calculations**
 - Models created using T-MATS were **simulated with constant and step inputs** and **results were within 2%** of those obtained from simulation of C-MAPSS, which is adequate for control design
- Future verification may include:
 - Simulation with a **better variable bleed valve *input*** model
 - Testing of the model with a **non-zero customer bleed**
 - **Closed-loop simulation** of the T-MATS model



Thanks.
Questions?



References

- 1 DeCastro, J. A., Litt, J. S., and Frederick, D. K., "A Modular Aero-Propulsion System Simulation of a Large Commercial Aircraft Engine," Proceedings of the 44th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit, AIAA 2008-4579, Hartford, CT, July 2008.
- 2 Frederick, D. K., DeCastro, J. A., and Litt, J. S., *User's Guide for the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS)*, NASA/TM-2007-215026, October 2007.
- 3 Liu, Y., Frederick, D. K., DeCastro, J. A., Litt, J. S., and Chan, W. W., *User's Guide for the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS), version 2*, NASA/TM-2012-217432, March 2012.
- 4 Csank, J., May, R. D., Litt, J. S., and Guo, T.-H., "Control Design for a Generic Commercial Aircraft Engine," Proceedings of the 46th AIAA/ASME/SAE/ASEE Joint Propulsion Conference, AIAA-2010-6629, Nashville, TN, July 2010.
- 5 May, R. D., Csank, J., Lavelle, T. M., Litt, J. S., and Guo, T.-H., "A High-Fidelity Simulation of a Generic Commercial Aircraft Engine and Controller," Proceedings of the 46th AIAA/ASME/SAE/ASEE Joint Propulsion Conference, AIAA-2010-6630, Nashville, TN, July 2010.
- 6 Chapman, J. W., Lavelle, T. M., May, R. D., Litt, J. S., and Guo, T.-H., *Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS) User's Guide*, NASA/TM-2014-216638, January 2014.



Appendix



Simulating the engine models: Solver setup

- T-MATS **steady-state** solver configured to **match solver in C-MAPSS**

Solver setting	C-MAPSS (D)IS	T-MATS SSIS
Max Jacobian perturbation	N/A	0.01
Max solution change per time-step (%)	N/A	2
Max iterations before Jacobian recalculation	N/A	100
Max iterations for while-loop	100	N/A
Termination condition (%)	2	1

- T-MATS **dynamic** solver configured in various ways to study **sensitivity** of results to changes in solver
 - Varying maximum number of iterations had **little effect**
 - Decreasing termination condition; increasing maximum Jacobian perturbation, maximum solution change/time-step **improved** matching

Solver setting	Variations in T-MATS DIS
Max Jacobian perturbation	0.001, 0.05 (0.0441 for SS1)
Max solution change per time-step (%)	0.5, 5, 10
Max iterations before Jacobian recalculation	15, 500, 1000
Max iterations for while-loop	100
Termination condition (%)	0.1, 2, 5

- Using these results, **dynamic iterative solver** was configured
 - Maximum Jacobian perturbation: 0.04
 - Maximum solution change/time-step: 5%
 - Maximum iterations before recalculating Jacobian: 100
 - Maximum while-loop iterations: 100
 - Termination condition: 0.01%



Simulating the engine models: Steady-state solver

- Comparison of **steady-state solver results** for four simulations
- Most are within 1% of each other
 - **Fan R-line** and **BPR** differ the **most** between models
 - **Relative accuracy of PR** and **shaft speeds** (N_f , N_c) observed in **relatively small mismatch** in flow characteristics at **turbine exits**
 - **Inaccuracy of fan R-line** and **inlet flow rate** (W_{20}) seen in **large mismatch in flow rates**, especially upstream of splitter

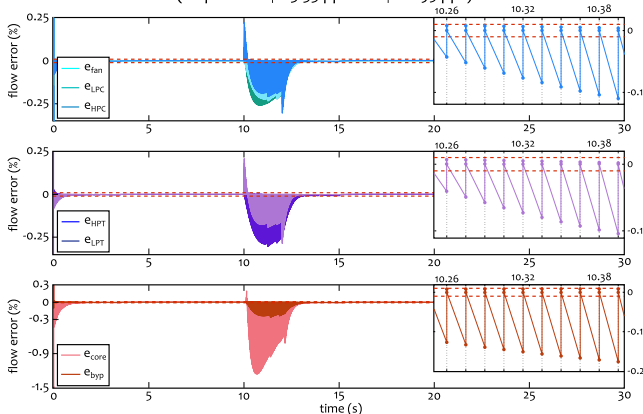
Variable	$W_f = 1.67$ pps	$W_f = 3.33$ pps	$W_f = 4.17$ pps	$W_f = 6.95$ pps
Fan R-line	4.2402%	3.3082%	3.1824%	2.6227%
LPC R-line	0.4115%	0.8505%	0.8454%	1.9311%
HPC R-line	1.3964%	1.1804%	1.5112%	2.5628%
HPT PR	0.0382%	0.0626%	0.0592%	0.0282%
LPT PR	0.7709%	0.654%	0.5419%	0.7358%
W_{20}	1.9178%	1.2927%	1.2474%	0.5532%
BPR	3.1275%	2.4064%	2.2044%	1.9136%
N_f (rpm)	0.7043%	0.311%	0.2625%	0.3203%
N_c (rpm)	0.2403%	0.2899%	0.3193%	0.369%
$F_{q, hyp}$ (lb _f)	2.6387%	1.5826%	1.3944%	0.0534%
$F_{q, core}$ (lb _f)	2.3795%	1.7056%	1.4905%	0.8556%



Simulating the engine models: Dynamic solver

- **Evolution of flow errors** shows how solver iterates the most during fuel transition to **maintain balanced flow** throughout the system

Dynamic T-MATS model: Flow error evolution through simulation
(step from $W_f = 3.33$ pps to $W_f = 6.95$ pps)





A Process for the Creation of T-MATS Propulsion System Models from NPSS Data

Jeffryes W. Chapman

Thomas M. Lavelle

Jonathan S. Litt

Ten-Huei (OA) Guo

T-MATS Workshop

Cleveland, OH

April 15, 2015



Acknowledgments

Funding for this work was provided by NASA Aviation Safety Program's Vehicle Systems Safety Technology Project (VSST)



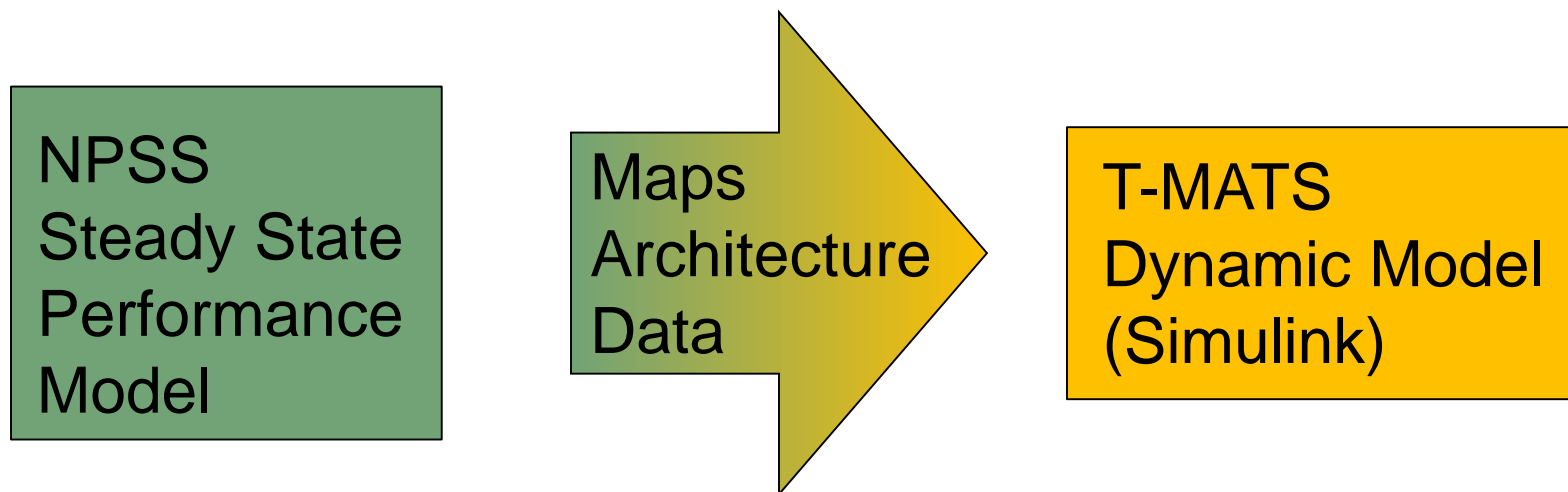
Outline

- Background
- Process
- Example
- Conclusion
- References and Download information



Background, T-MATS

- **Toolbox for the Modeling and Analysis of Thermodynamic systems, T-MATS**
 - Modular thermodynamic modeling framework
 - High fidelity dynamic gas turbine modeling capability based around component maps
 - Built in MATLAB/Simulink utilizing S-functions written in C
 - Open source and completely modifiable
- **Case study: Selecting T-MATS for generation of a dynamic model from an NPSS performance model.**





Background, Modeling example

- When developing a new engine, control design is typically performed after engine cycle design and performance analysis, and requires a dynamic engine model.
- Example situation:
 - Numerical Propulsion System Simulation (NPSS) used for cycle design and performance analysis
 - MATLAB®/Simulink® used for gas turbine control system development
- How are plant models for gas turbine control system development obtained?
 - Develop a new model from engine performance specifications.
 - Leverage pre-existing plant model directly by integrating it with the control system development tool (e.g., wrap NPSS into a Simulink S-function).
 - Develop a new model based on the pre-existing model.



Model creation

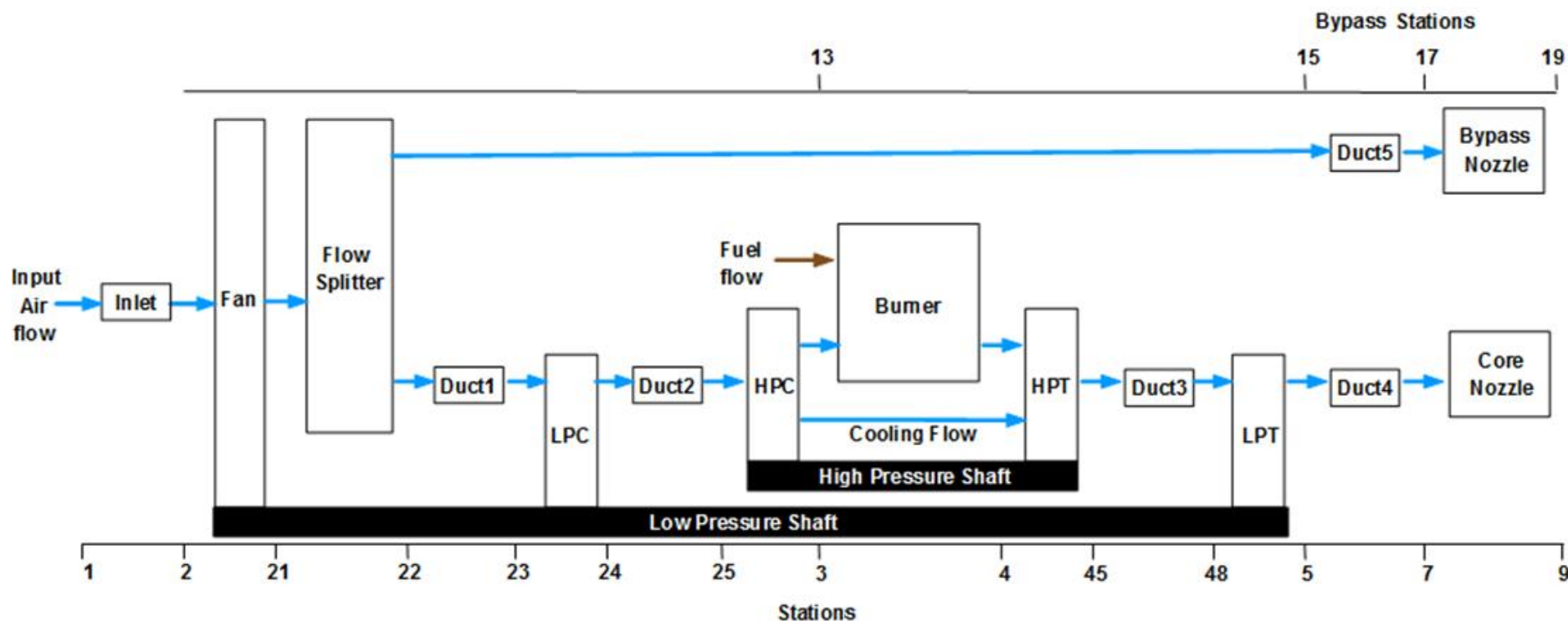
- ### Process Overview

1. Gathering inputs and making modifications, verify the models have the same modeling strategy and inputs
 - a) Convert maps and constants from NPSS into a usable format for T-MATS
 - b) Modify T-MATS components to be compatible with the new maps and constants
2. Component level testing, verify component models are operating similarly
3. System level testing without a solver, verify simulation connections
4. Steady state system level testing, verify the system converges to the correct operating points
5. Converting the steady state simulation to a dynamic simulation

JT9D engine, model

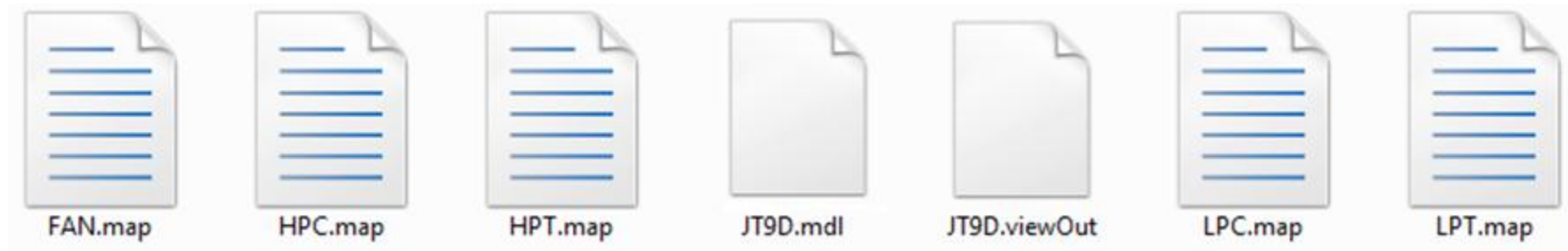
- Engine Example:

- Publicly available dual spool high-bypass turbofan engine model
 - Implemented in NPSS then converted to T-MATS
 - Plant components and architecture representative of both models and utilized in steps 1-3 from the previous slide.



JT9D Engine: Model Matching Files

- NPSS files required for model matching:
 - Compressor and turbine map files (*.map)
 - NPSS output data file (*.viewOut) should contain
 - Operating point dependent variables
 - Thermodynamic properties (e.g. Pt, Tt, W, etc.) at each engine stage
 - Environmental variables as well as performance variables such as thrust
 - NPSS model definition (*.mdl)
 - Additional component variables not detailed in the data file, e.g. LHV
 - Static values consistent across the envelope





JT9D Engine, Operating Points

- NPSS data utilized:

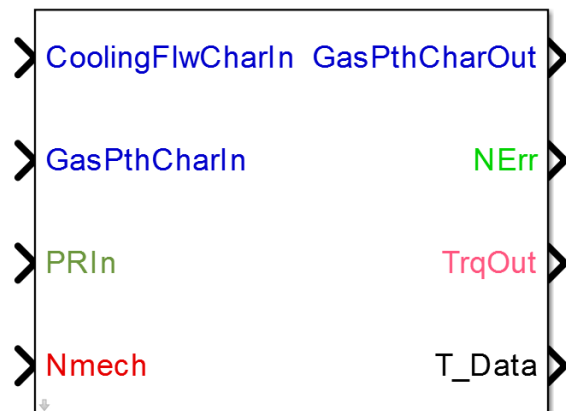
- Two distinct operating points were run in NPSS to generate data for the model matching
 - Model inputs and modification requirements (step 1) were determined based on the takeoff operating point.
 - Component and system level testing (step 2 and 3) were completed using the takeoff operating point.
 - Steady state and dynamic model testing (step 4 and 5) was completed with both operating points. Additionally, an alternative method of automatically generating model matching scale factors (T-MATS iDesign tool) was implemented to demonstrate tool feasibility.

operating point	altitude, ft	Mach Number	Ambient Temperature, degR	fuel flow, pps
design point or cruise	34000.0	0.8	448.43	1.91
off-design point or takeoff	0.0	0.0	545.67	5.0

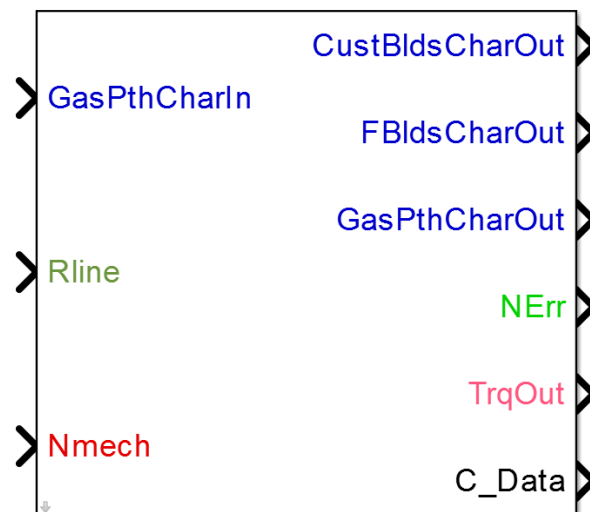


T-MATS Maps and Constants

- Baseline NPSS uses almost all maps and constants in the same way as T-MATS.
 - Turbine maps based on PR lookup
 - Compressor maps based on R-line
 - Key components input similarity:
 - Nozzle throat area used for thrust calculation
 - duct pressure drop
 - burner LHV and efficiency
 - Turbine cooling flow before or after OD turbine component
 - Fractional calculated compressor bleed flow



Turbine



Compressor



Key Modifications: Scale Factors

- **NPSS and T-MATS both scale inputs and outputs to compressor and turbine maps**
 - Typically used to shrink or stretch a generic model or convert inputs or outputs to ratio values
 - Here, scale factors were used to ensure a model match
 - T-MATS scale factors are generated from NPSS output data by dividing a performance value by a map value. Alternatively the T-MATS tool iDesign was utilized to automatically generate scale factors from operating point performance data.
- **Corrected turbine speed**
 - T-MATS scales corrected shaft speed in the turbine by a constant (standard day temperature), while NPSS does not. This difference can be taken into account with the map scale factors.

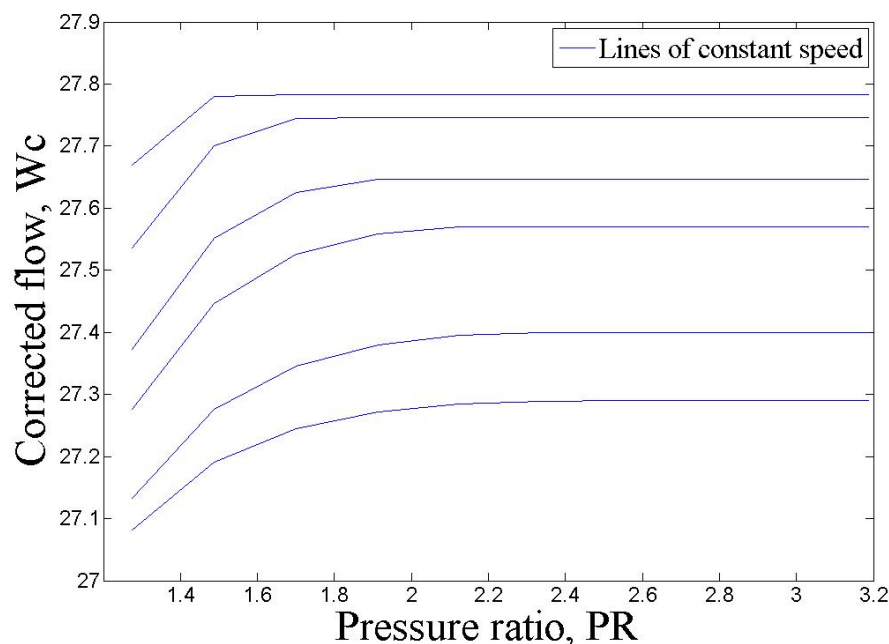
Component	map scale factor	equation
Compressor	s_Nc	$s_Nc = NcPerf / NcMap$
Turbine	s_Nc	$s_Nc = NcPerf \times \sqrt{T_std} / NcMap$
Compressor or Turbine	s_Wc	$s_Wc = WcPerf / WcMap$
	s_PR	$s_PR = (PRPerf - 1) / (PRMap - 1)$
	s_Eff	$s_Eff = EffPerf / EffMap$



Key Modifications: Turbine flow definition

- Flow Definition for turbine maps

- Baseline NPSS defines turbine map W_c as simply the turbine input flow
- T-MATS defines the turbine map W_c as the turbine input flow and a portion of the cooling flow

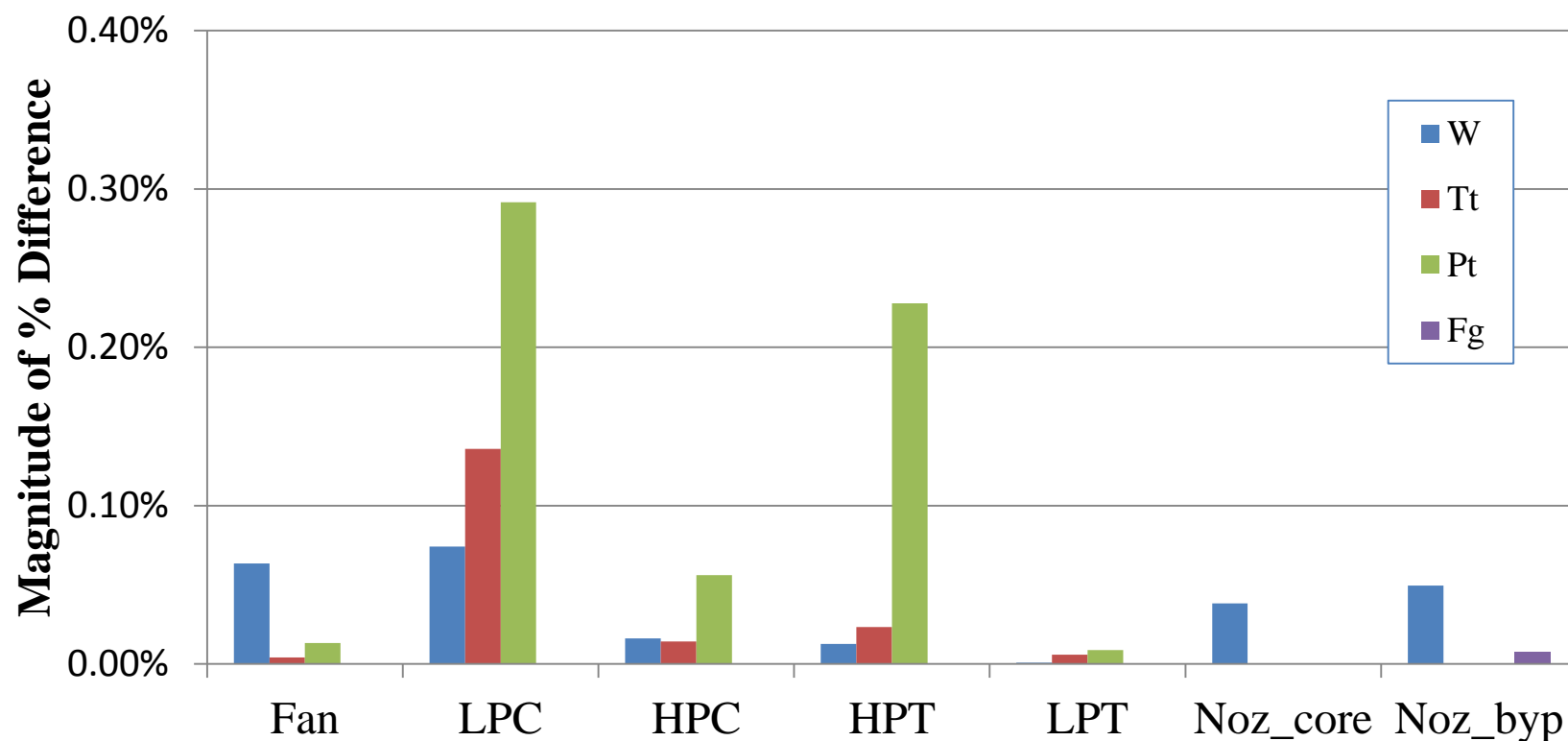


For T-MATS to use NPSS turbine maps, cooling flow must be removed from the T-MATS turbine map definition.



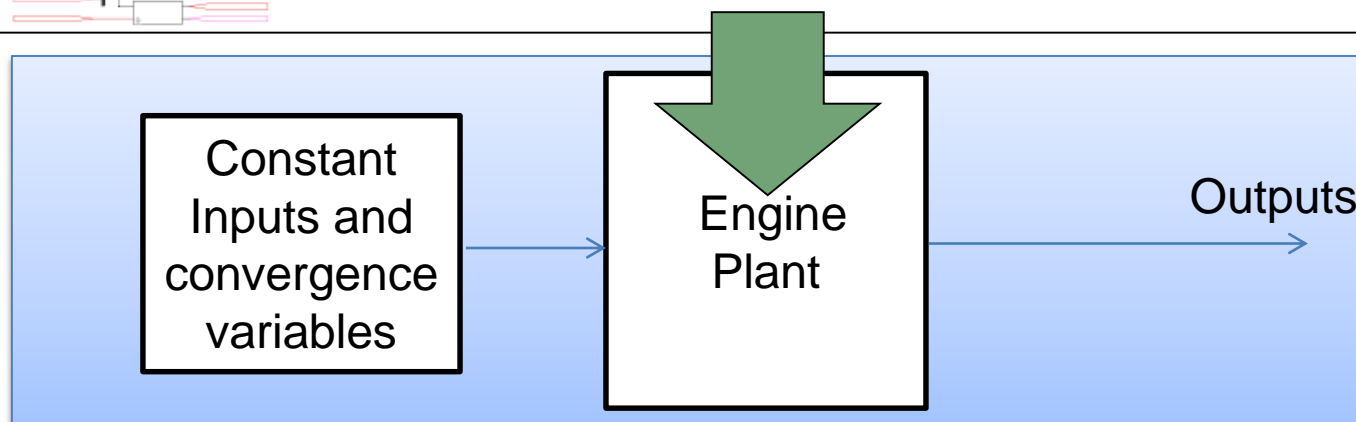
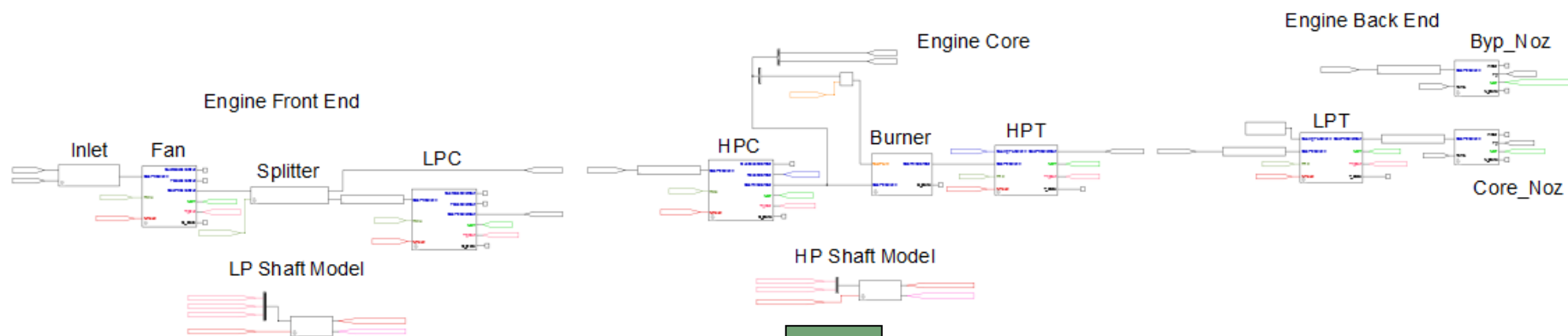
Component Level Matching

- **Component models built for each major model section**
 - NPSS station data at takeoff are used as inputs
 - Constant inputs without solver do not guarantee conservation of mass
 - Verifies component matching within acceptable limits



System Level Model

- Components combined to create a system plant model
 - Constant inputs without a solver do not guarantee conservation of mass
 - Inputs include control system inputs as well as convergence variables such as R-line.

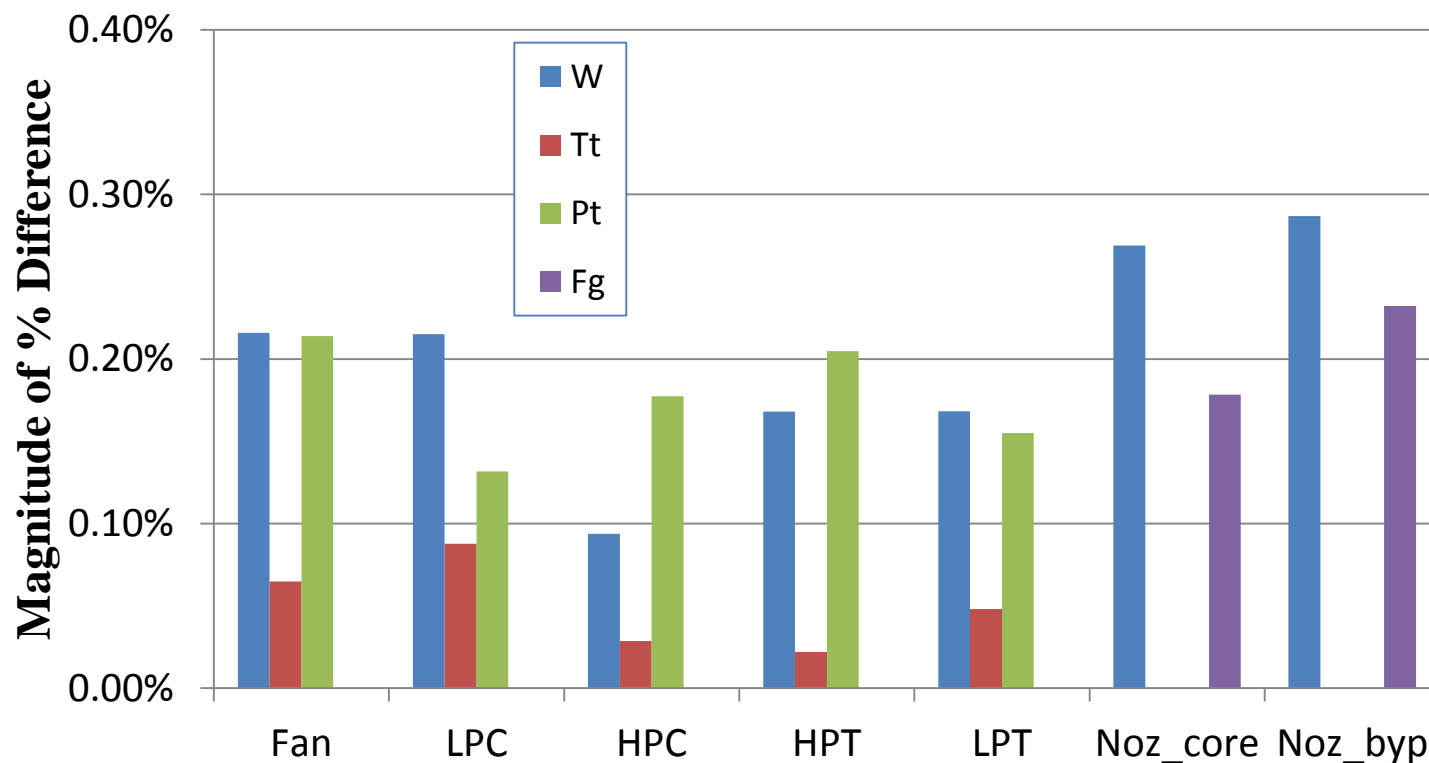




System Level Matching

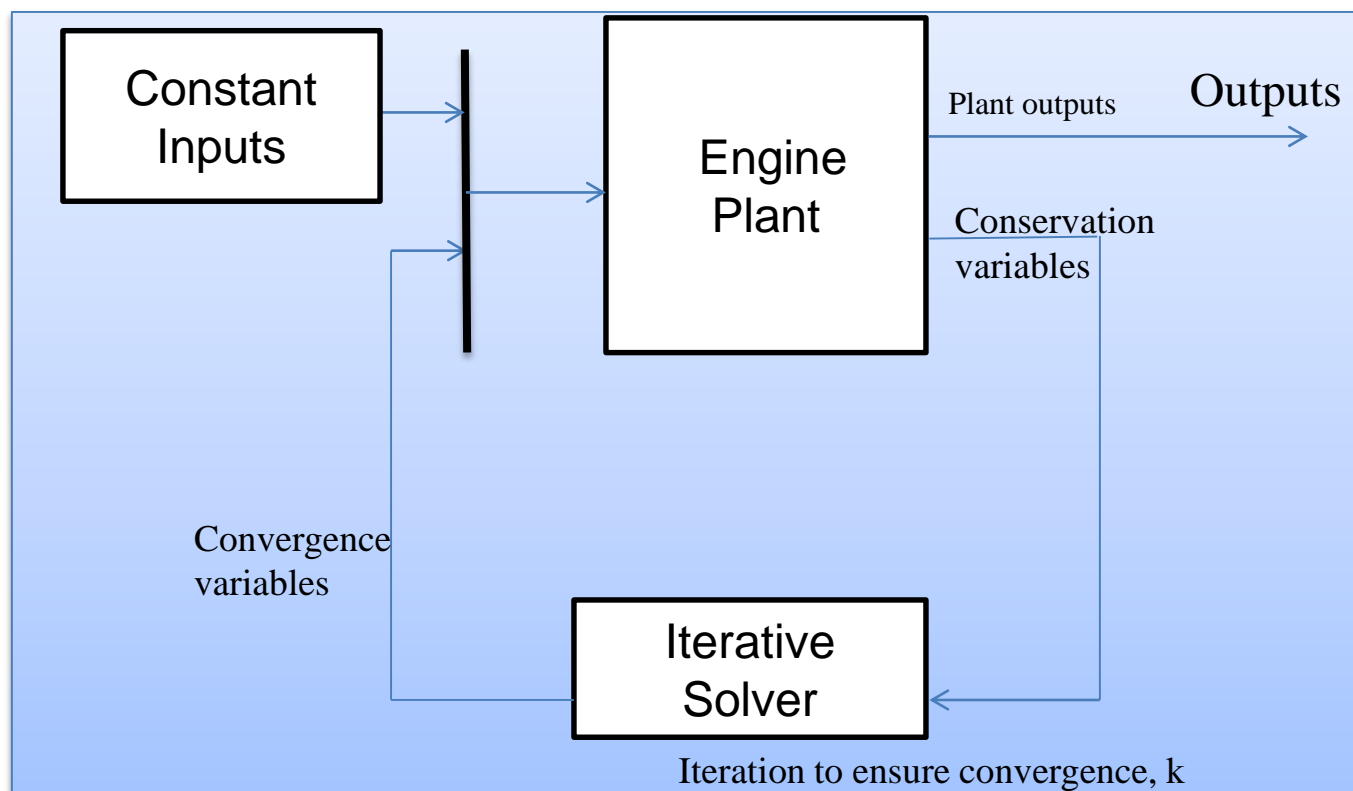
- System model matching

- Difference values generally higher than component matching due to error compounding, but still within acceptable levels.
- Verifies component connections are accurate



Steady State Model

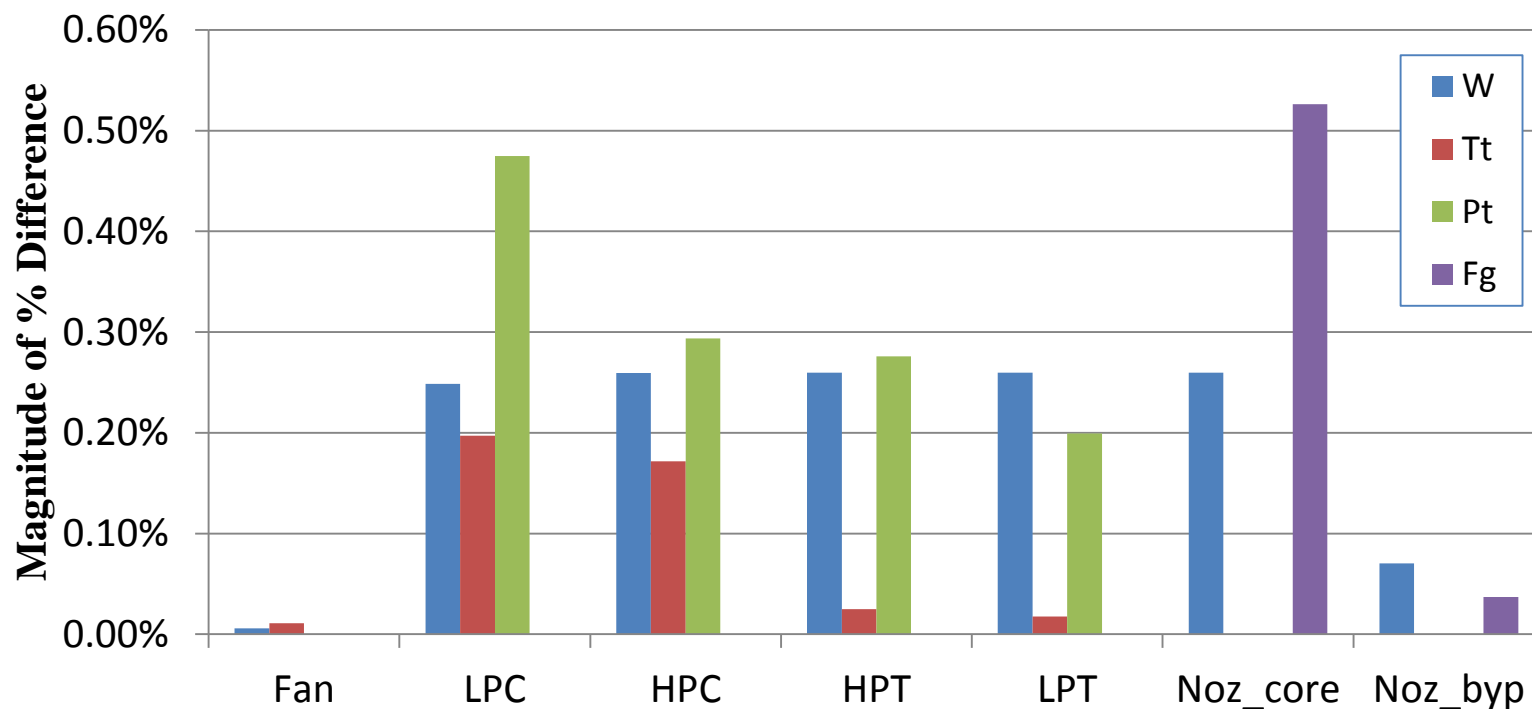
- Solver added to system to ensure conservation
 - Convergence variables used to drive conservation variables to zero
 - Constant inputs include envelope point and control inputs





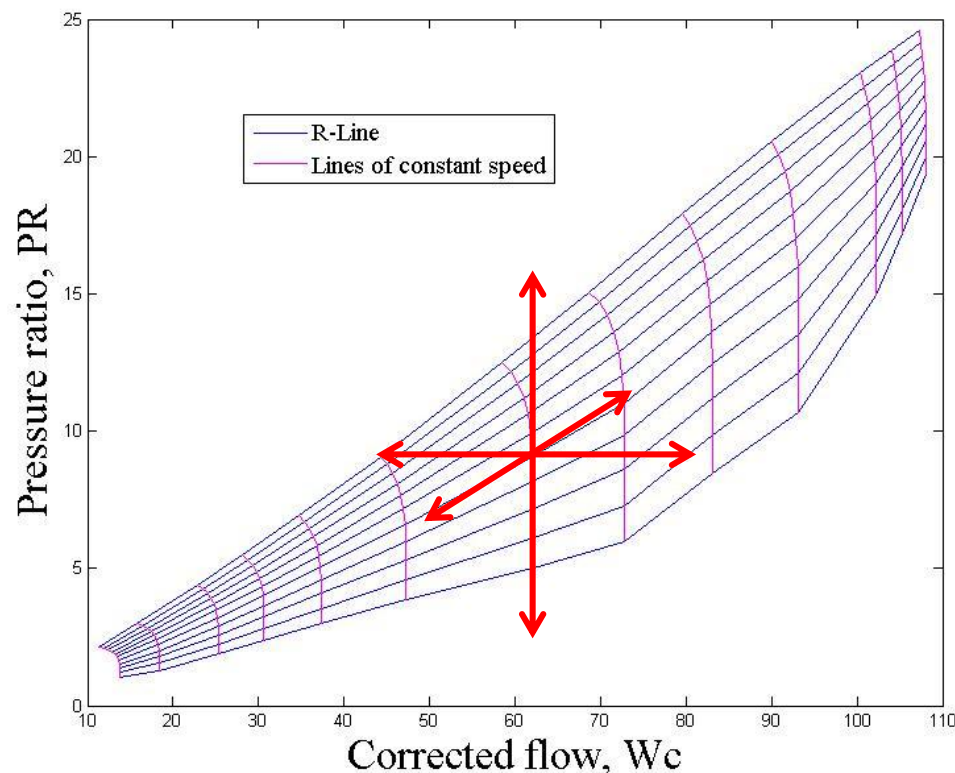
Steady State Model Matching

- Solver added to system model to ensure conservation of mass and negligible shaft acceleration
 - Fuel flow and environmental variables set to constants
 - Difference values higher than component matching, but still within acceptable levels.
 - Verifies system converges to correct operating point (in this case takeoff)



iDesign Tool

- iDesign tool develops map scale factors.
 - Uses scale factors to shift compressor maps, turbine maps, and nozzle throat areas effectively re-sizing components to fit a chosen operating point (design point)
- Compressor Map example:
 - PR at a given point modified by augmenting:
 - PR scale factor
 - W_c scale factor
 - Speed scale factor



*It should be noted that the iDesign tool will fit the model to whatever operating point is specified, which may mask modeling discrepancies. This issue may be mitigated by testing many operating points across the entire flight envelope.



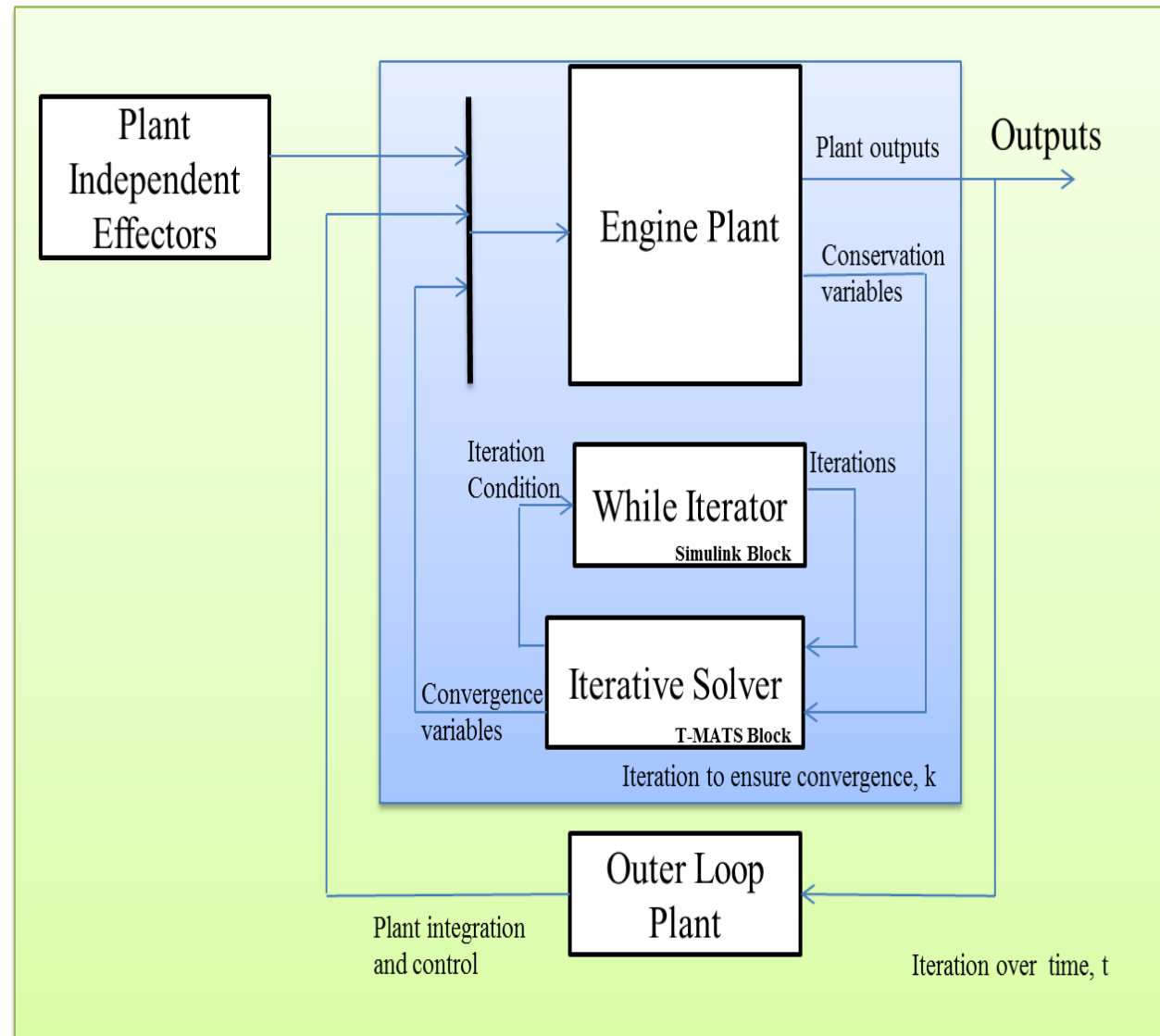
Alternate Operating Points and Constant Generation

- iDesign tool was run with the steady state model to automatically generate map scale factors to match the cruise operating point.
 - Steady state matching of two operating points (cruise and takeoff) were compared.
 - Average difference magnitudes comparable between scale factor generation methods
 - Additional operating point simulation verifies the match in alternate envelope conditions

Simulation Level	Solver	Operating Point	Scale Factor Derivation Method	Average Difference
Component	No	takeoff	NPSS derived	0.0550%
System	No	takeoff	NPSS derived	0.1558%
System	Yes	takeoff	NPSS derived	0.1891%
System	Yes	cruise	NPSS derived	0.1233%
System	Yes	takeoff	iDesign	0.2490%
System	Yes	cruise	iDesign	0.0910%

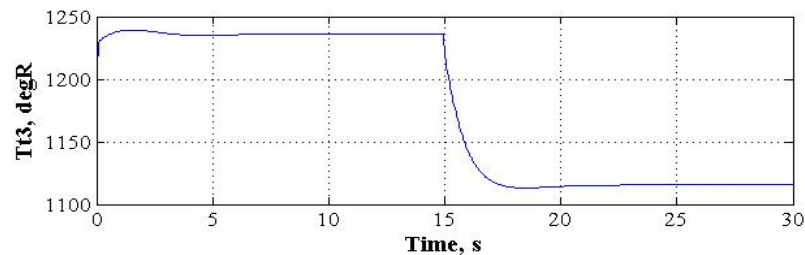
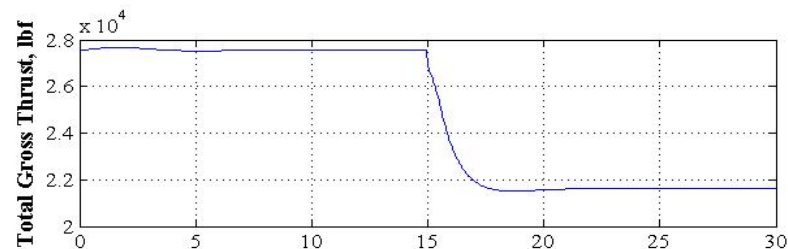
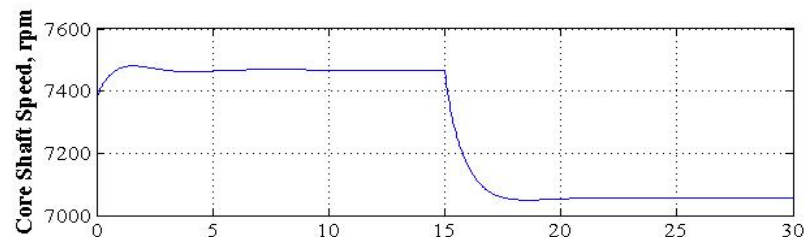
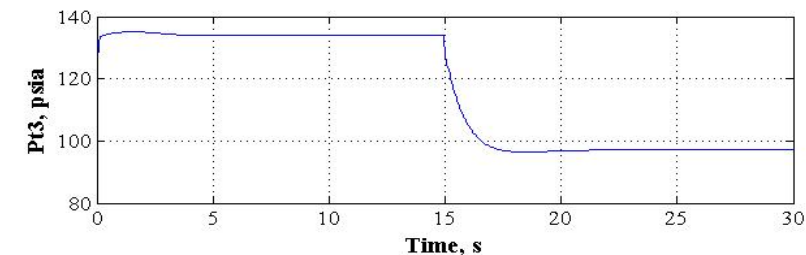
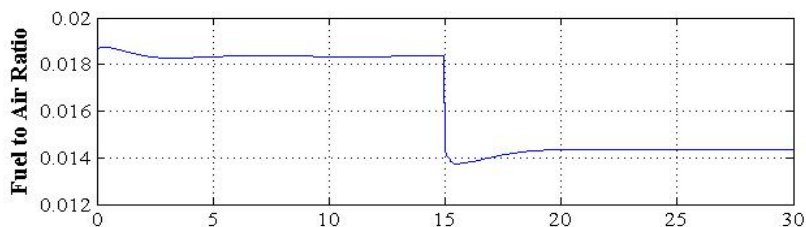
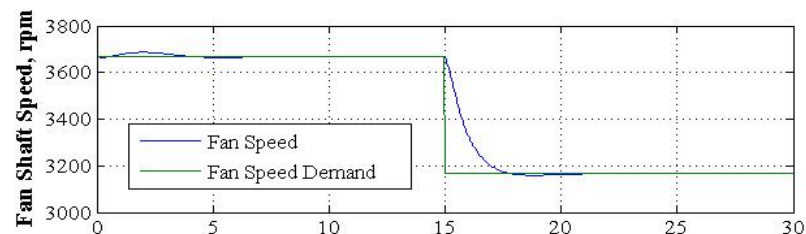
Dynamic Model

- System updated to dynamic simulation architecture
 - integrators used to determine shaft speed
 - convergence errors solved for at every time step
 - Shaft dynamic properties assumed based on engine class



Dynamic operation

- Simulation of “chop” (sudden drop in fan speed demand) maneuver
 - PI Fan speed controller designed for demonstration purposes
 - Fan Speed Demand drops at 15 seconds with fall time roughly 2 seconds
 - Results typical for a dual spool high bypass turbofan





Conclusions

- A simple process may be followed to derive a T-MATS model from a baseline NPSS model.
 - Minimal Modifications to the T-MATS block set must be performed to achieve a “good” matching.
 - Scale factors may be generated manually from NPSS data or automatically with the T-MATS tool iDesign and operating point data.
 - Model may be updated to run dynamically by performing a quick adjustment to the model architecture.
- T-MATS enables rapid dynamic model creation and eliminates cross-platform model integration when system components are built in Simulink



References and Download Information

- Download information may be found at:
<https://github.com/nasa/T-MATS/releases/>

- References:

1. Chapman, J.W., Lavelle, T.M., May, R.D., Litt, J.S., and Guo, T.M., “Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS) User’s Guide,” NASA/TM-2014-216638, January 2014.
2. Chapman, J.W., Lavelle, T.M., May, R.D., Litt, J.S., Guo, T-H., “Propulsion System Simulation Using the Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS),” 2014 AIAA Joint Propulsion Conference, Cleveland, OH, Jul 28-30, 2014.
3. Lavelle, T.M., Chapman, J.W., May, R.D., Litt, J.S., and Guo, T.H., “Cantera Integration with the Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS),” 2014 AIAA Joint Propulsion Conference, Cleveland, OH, Jul 28-30, 2014.
4. Chapman, J.W., Lavelle, T.M., Litt, J.S., Guo, T-H., “A Process for the Creation of T-MATS Propulsion System Models from NPSS Data,” 2014 AIAA Joint Propulsion Conference, Cleveland, OH, Jul 28-30, 2014.
5. Zinnecker, A.M., Chapman, J.W., Lavelle, T.M., and Litt, J.S., “Development of a twin-spool turbofan engine simulation using the Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS),” 2014 AIAA Joint Propulsion Conference, Cleveland, OH, Jul 28-30, 2014.



Cantera Integration with the Toolbox for Modeling and Analysis of Thermodynamic Systems (T-MATS)

Thomas Lavelle
NASA GRC



Team

- **Thomas M. Lavelle**

NASA Glenn Research Center, Cleveland, OH 44135

- **Jeffryes W. Chapman**

Vantage Partners, LLC. Cleveland, OH 44135

- **Ryan D. May**

previously with Vantage Partners, LLC. Cleveland, OH 44135

- **Jonathan S. Litt**

NASA Glenn Research Center, Cleveland, OH 44135

- **Ten-Huei Guo**

NASA Glenn Research Center, Cleveland, OH 44135



Goal

- Increase flexibility of T-MATS
- Cantera increases flexibility of thermodynamics
 - Can model any flow
- M-file elements allow users to prototype engineering elements
- To execute not too much slower than standard T-MATS



T-MATS

- Simulink code
- Library of thermodynamic elements
 - Standard library includes elements typical of aeropropulsion
- Newton Raphson solver
- Default thermodynamic table is air, water, and a hydrocarbon fuel
- Systems can be modeled outside the standard elements/thermo
 - Create new thermo tables
 - Create elements



Cantera

- Object-oriented software tools for problems involving chemical kinetics, thermodynamics, and/or transport properties
- C++ based code with interfaces for python, matlab, C, and fortran 90
- <https://code.google.com/p/cantera/>



Integration of T-MATS with Cantera

- Allows any fluid combination to be modeled
- Specify the thermodynamics of the possible products
 - Similar to CEA thermo.inp file
- Requires specification of all “reactants” for the simulation
 - Similar to CEA reactant cards
 - Specify the different possible starting flows by composition



```
Species = { .7547 .232 .0128 0 0 0;
            1 0 0 0 0 0;
            .922189 .077811 0 0 0 0 0;
            0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0};

Name = { 'N2' 'O2' 'AR' " " ";
        'H2O' " " " " " "; 'CH2' 'CH' " " " " ";
        " " " " " " ; " " " " " " ; " " " " " " };
```

- Species and Name arrays need to be defined
- A model with this definition can run with mixtures of Air, Water, and JP-7
- Allows for models of aircraft engines with humidity



T-MATS Cantera Fluid Arrays

Information	Index	Description
W	1	Weight of the flow
Tt	2	Total temperature
Pt	3	Total pressure
ht	4	Total enthalpy
comp1 (to) comp10	5-14	Percentage of flow composition for reactants 1 to 10
s	15	Entropy
rhot	16	Total density
Ts	17	Static temperature
Ps	18	Static pressure
hs	19	Static enthalpy
rhos	20	Static density
Vflow	21	Flow velocity
MN	22	Flow Mach number
A	23	Flow area
gamt	24	Total gamma
gams	25	Static gamma

- Each fluid location in a thermodynamic model is represented by an array that contains all the fluid properties at a given location



T-MATS Cantera Fluid Functions

Function	Description
<code>add(flow1, flow2)</code>	Add <code>flow1</code> and <code>flow2</code> together, conserving enthalpy and mass
<code>copyFlow(flow)</code>	Copy the information from <code>flow</code> to another flow
<code>getMassFraction(flow, c)</code>	Return the mass fraction of compound <code>c</code> in the object <code>flow</code>
<code>set_hP(flow, ht, Pt)</code>	Set the total conditions based on <code>flow</code> , total enthalpy and total pressure
<code>set_MN1(flow)</code>	Set the static conditions to sonic based on flow conditions
<code>set_MNPs(flow, Ps)</code>	Set the static conditions based on <code>flow</code> and input static pressure
<code>set_SP(flow, S, Pt)</code>	Set the total conditions based on <code>flow</code> and input entropy and total pressure
<code>set_TP(flow, Tt, Pt)</code>	Set the total conditions based on flow, total temperature and total pressure
<code>set_TsPsmN(flow, Ts, Ps, MN)</code>	Set the conditions based on <code>flow</code> , static temperature, static pressure and Mach

- All communication between Cantera and T-MATS is handled by these functions
- Functions return a new Cantera Fluid Array based on inputs (see previous slide)



T-MATS Element Files

- Library of standard elements released in Simulink m-file format
- Allows for development and prototyping
- Elements are interpreted
 - No need to compile
- Engineers can quickly create new elements
- Block sets are released with T-MATS Cantera package



Instance Information

- Needed a way to store instance information from one pass to another
- Created two functions to store and retrieve information from one pass to another
 - Variables are stored in the MATLAB workspace with the object instance name attached to the variable instance name
- `setV` sets the value of a variable in the workspace
- `getV` gets the value of a variable from the workspace

```
path = stripchar( gcb() );  
setV( 's_C_Nc', path, s_C_Nc );  
s_C_Nc = getV( 's_C_Nc', path );
```



Some Examples from Compressor Element

- Setting the exit conditions

```
FOideal = set_SP( FI, FI(s),PtOut );  
htOut = FI(ht) + ( FOideal(ht) - FI(ht) )/eff;  
% set the exit conditions to known enthalpy and  
%pressure  
FO = set_hP( FI, htOut, PtOut );
```

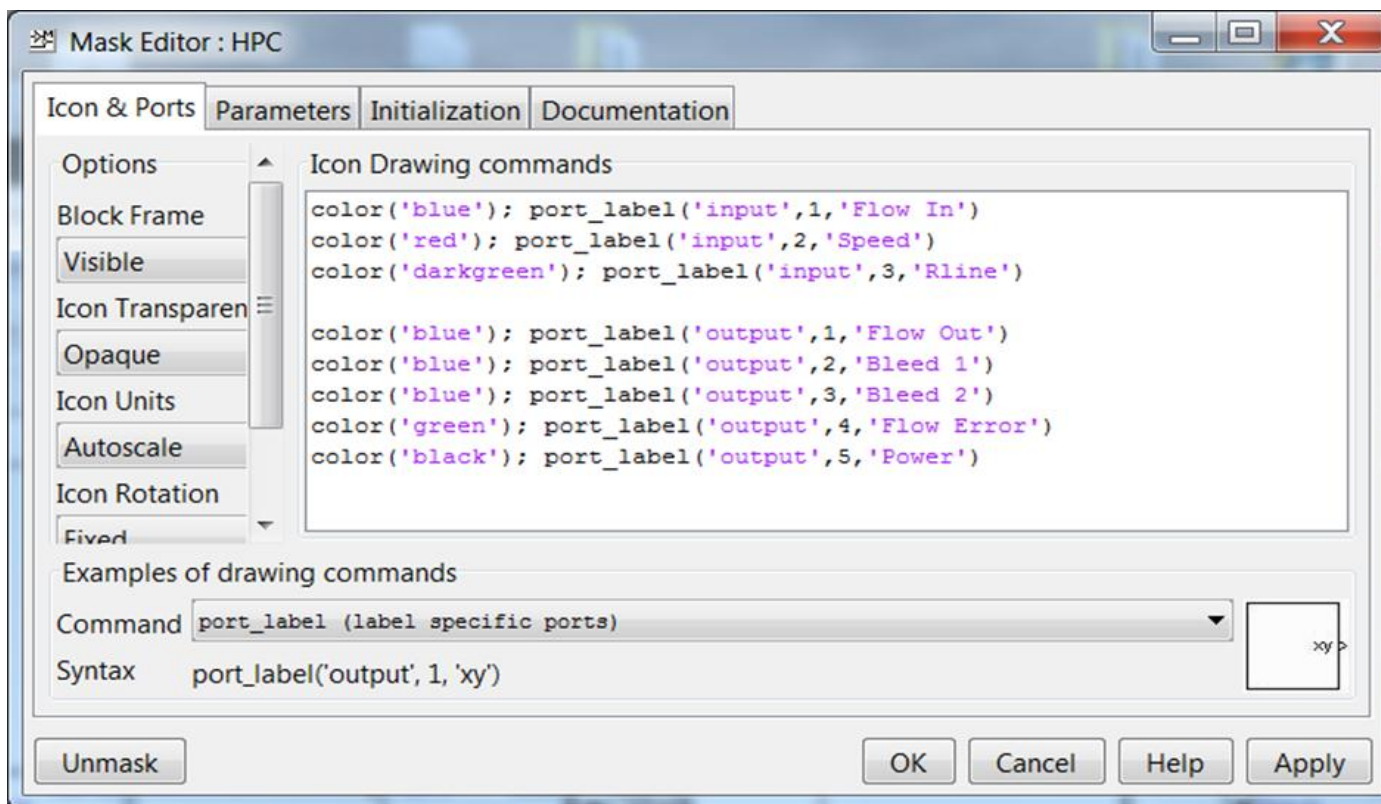


Some Examples from Compressor Element

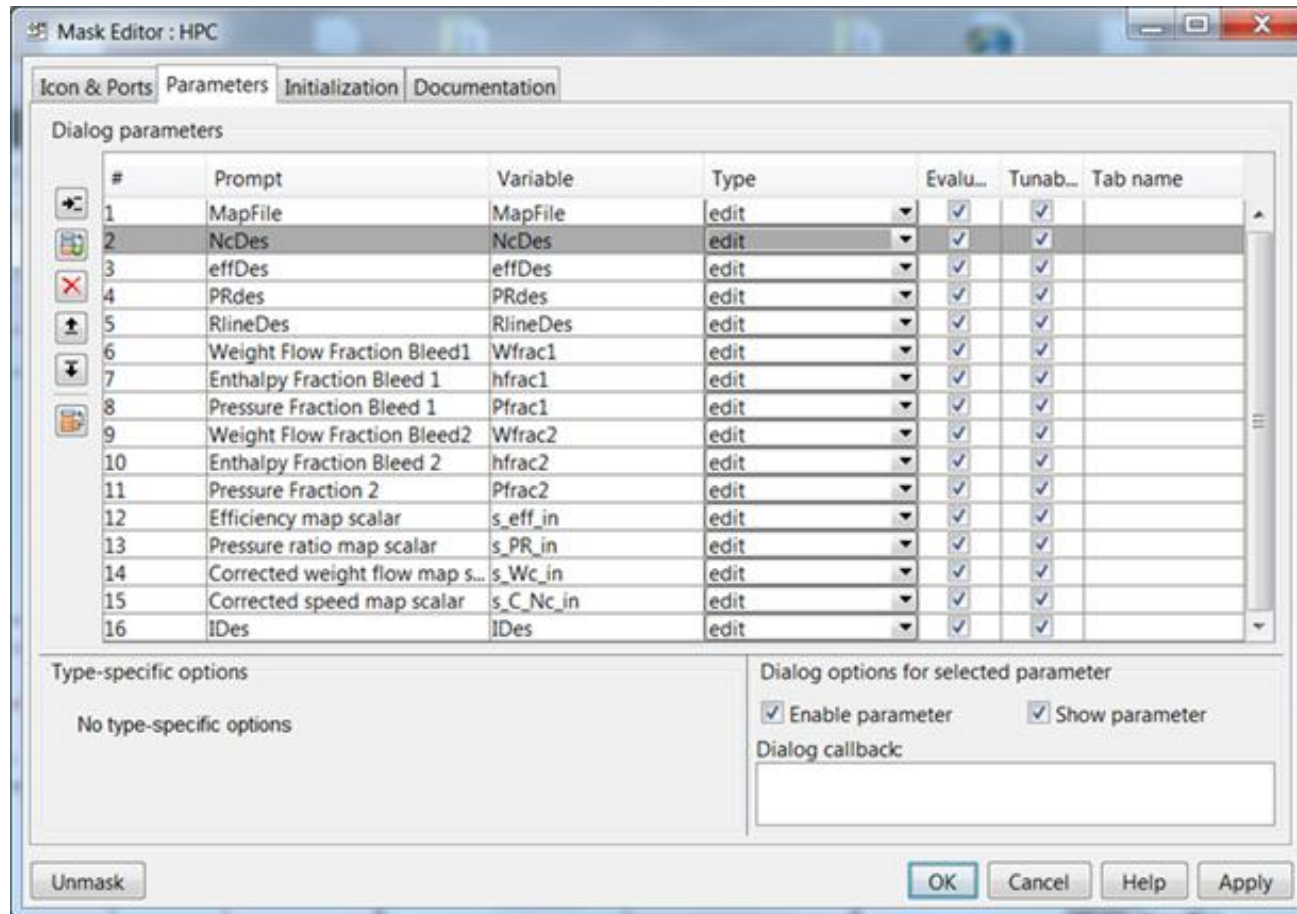
- Design Point Scaling

```
if IDes < .5
    s_eff = effDes / effMap;
    s_PR = ( PRdes - 1 )/( PRmap - 1 );
    s_Wc = WcIn/ WcMap;
    setV( 's_eff', path, s_eff );
    setV( 's_Wc', path, s_Wc );
    setV( 's_PR', path, s_PR );
elseif IDes < 1.5
    % get the maps scalars from the workspace
    s_eff= getV( 's_eff', path );
    s_Wc= getV( 's_Wc', path );
    s_PR= getV( 's_PR', path );
else
    % use the input values
    s_eff = s_eff_in;
    s_Wc = s_Wc_in;
    s_PR = s_PR_in;
end
```

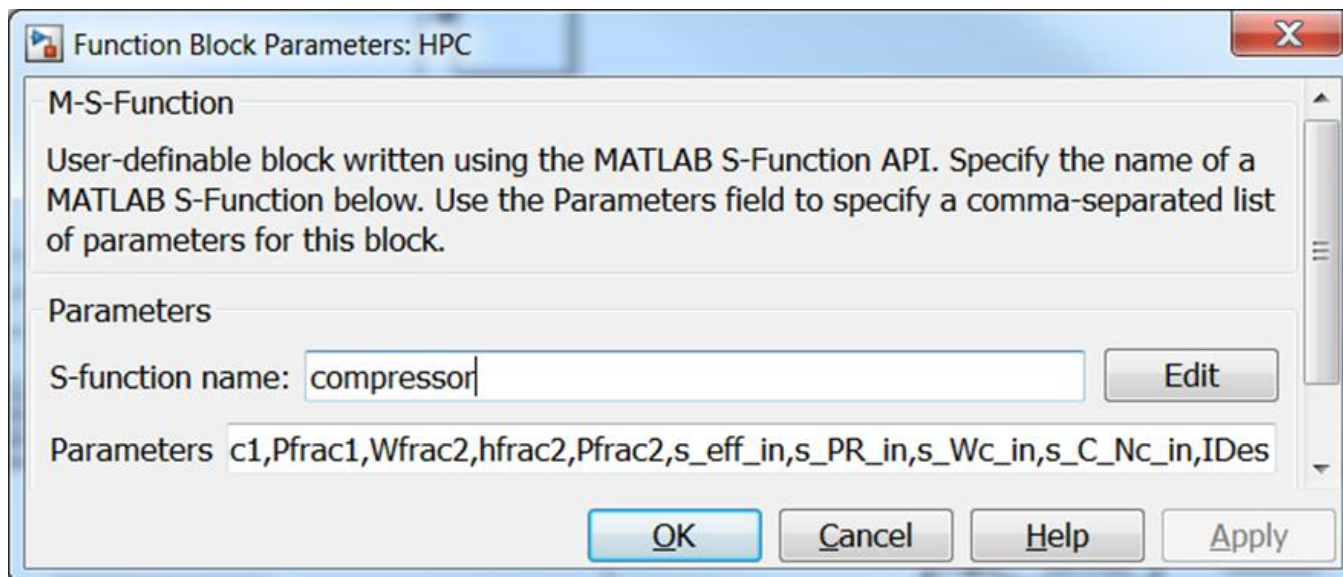
Simulink Objects



- Mask appearance
- Describes port labels and colors
- Label colors are standard based on T-MATS style

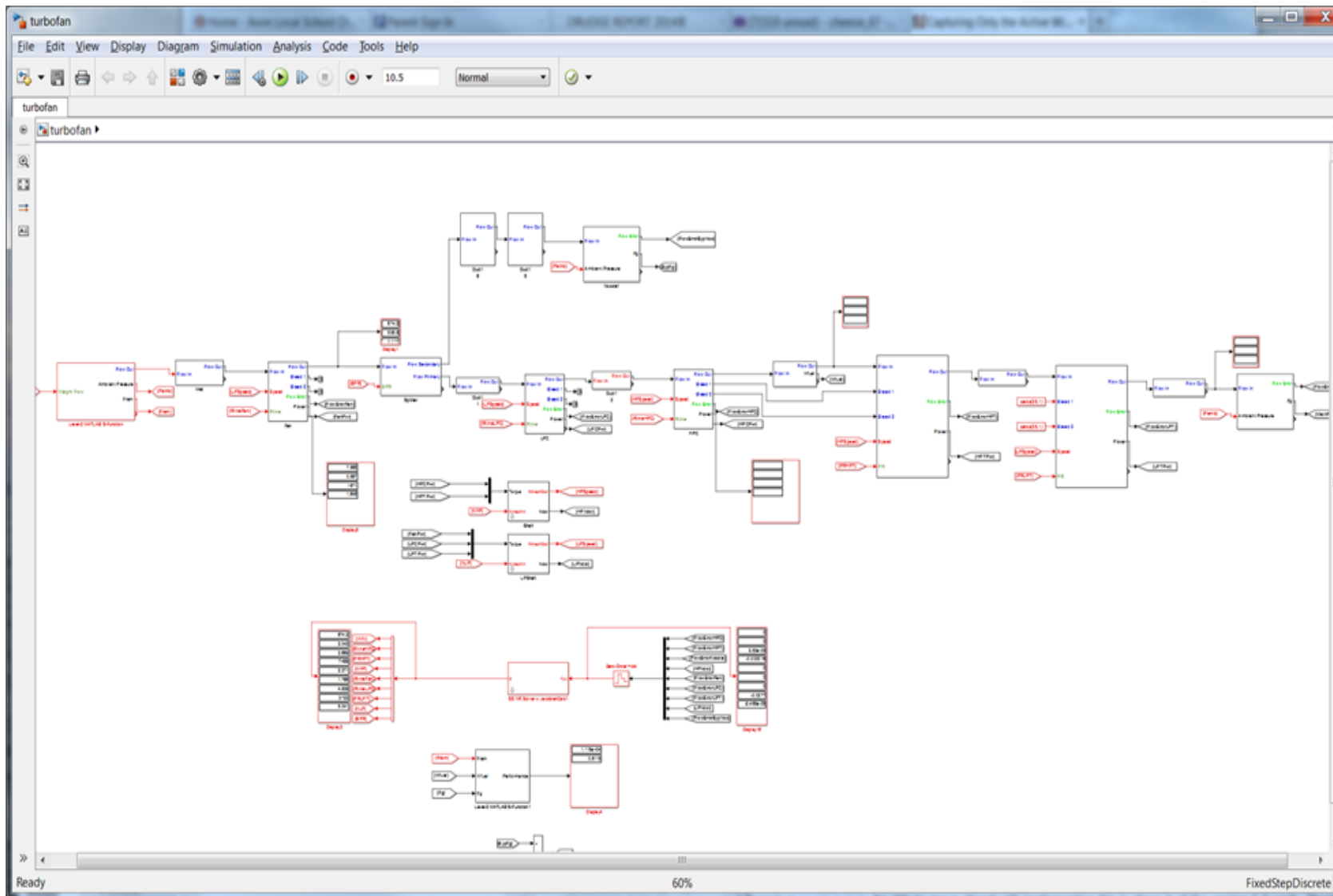


- Parameter list
- Lists the variables that can be input by the user to the dialog box



- S-function block parameters
- Utilizes m-file to create S-function
- Maps parameter dialog box to m-file

Turbofan Model –JT9D

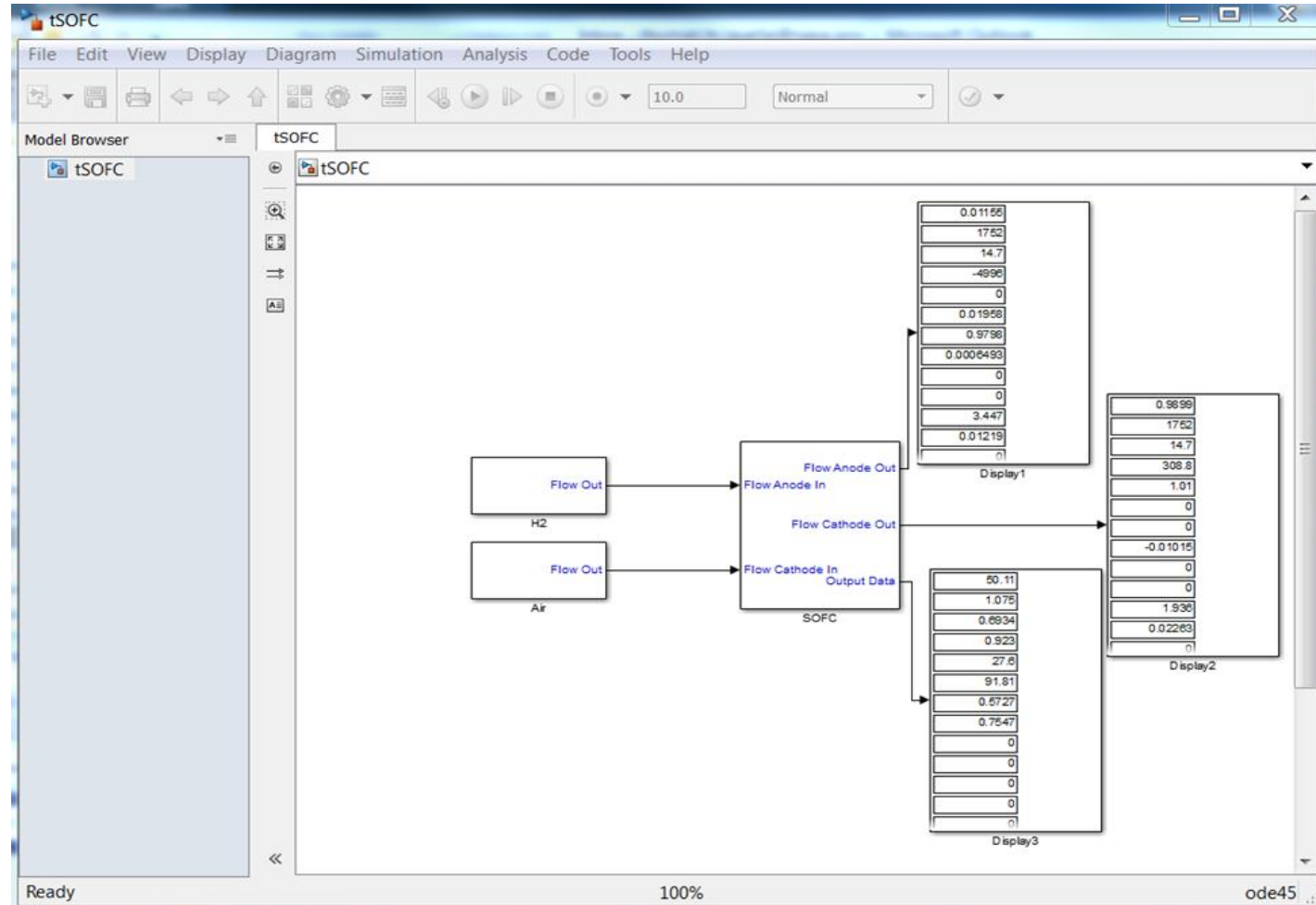




Turbofan Model –JT9D

	NPSS with JANAF Output	TMATS Cantera Output
Altitude	34000 ft	34000 ft
Mach number	.8	.8
Weight flow	674 lbm/sec	674 lbm/sec
Thrust	11194 lbf	11182 lbf
SFC	.6113	.6116

Fuel Cell Model



- Reactants are Air, H2, O2, and H2O



Fuel Cell Model

Specifying the reactants:

```
Species = { .7547 .232 .0128 0 0 0;  
            1 0 0 0 0 0;  
            1 0 0 0 0 0;  
            1 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0};  
Name = { 'N2' 'O2' 'AR' " " " "  
        'H2' " " " " " " " "  
        'H2O' " " " " " " " " " " " " }
```

Getting the mass fractions of an element:

```
xN2_cOut = getMassFraction( FI_O2, 'N2' );  
xO2_cOut = getMassFraction( FI_O2, 'O2' );
```



Fuel Cell Model

Removing oxygen from the flow:

$xO2_Cathode1 = getMassFraction(FI_Cathode1, 'O2')$

%Composition as mass flow (g/sec)

*$wO2_Cathode1 = xO2_Cathode1 * w_Cathode1$*

%Composition as molar flow rate (mol/sec)

$M_O2_Cathode1 = wO2_Cathode1 / 32.$

%Calculates composition after electrochemistry...

*$M_O2_Cathode2 = M_O2_Cathode1 - ((M_H2_Anode1 / 2.0) * pctH2util);$*

*$M_O2_Cathode2 = M_O2_Cathode1 - ((M_H2_Anode1 / 2.0) * pctH2util);$*

*$wO2_lost = (M_O2_Cathode1 - M_O2_Cathode2) * 32. * 0.002205 \text{ \% lb/sec}$*

$FI_tempO2(8) = 1;$

$FI_tempO2(W) = -wO2_lost;$

$FI_tempO2 = set_TP(FI_tempO2, FI_Cathode2(Tt), FI_Cathode2(Pt));$



Conclusion

- Cantera has been integrated with T-MATS
 - Capable of modeling any thermodynamic flow
- Simulink block sets and MATLAB m-files
 - Allows for prototyping
- Greatly increases the flexibility of T-MATS
- Slower than standard T-MATS



- Download information may be found at:
<https://github.com/nasa/T-MATS/releases/>

T-MATS Simulation of engine performance during Ice Particle Ingestion

Jeffryes Chapman

*T-MATS Workshop
Cleveland, OH
April 15, 2015*

Background

- In an effort to gain more insight into turbofan icing a dual spool turbofan engine was tested at the Propulsion Systems Lab (PSL) at NASA Glenn Research Center
 - The engine tested is obsolete and a dynamic simulation of the engine is not readily available
- PSL Testing summary:
 - Baseline steady state power
 - ice crystal cloud injected into the engine airstream
 - Ice ingestion resulted in accretion of ice in the LPC and in some cases resulted in loss of engine power.
- Goal:
 - Implement a real time algorithm for icing detection.
- Initial Steps:
 - Create dynamic engine model that simulates baseline operation and operation during ice cloud ingestion and subsequent icing at the cruise point.
 - Test algorithms designed to detect icing.

Simulations:

- All modeling was based on the work done under the Atmospheric Environment Safety Technology Project (AEST) and detailed in:

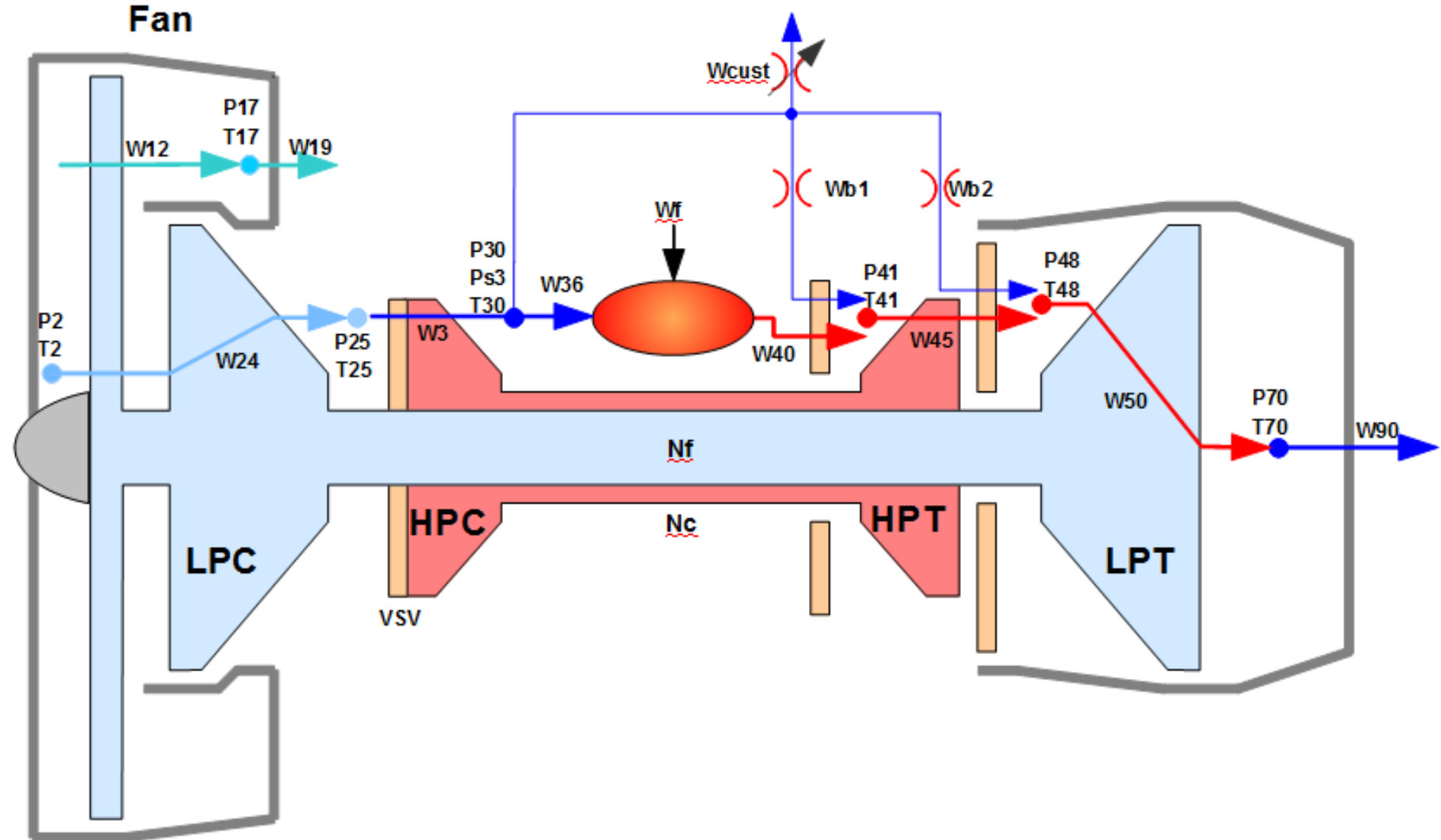
Jorgenson, P.C.E., Veres, J.P., Jones, S.M., "Modeling the Deterioration of Engine and Low Pressure Compressor Performance During a Roll Back Event due to Ice Accretion," AIAA Joint Propulsion Conference, Cleveland, OH, July 28-30, 2014.

T-MATS modeling effort:

- 2 engine simulations were created
 - Baseline engine model matched to steady state performance data using Maps provided by the engine manufacturer and scaled with the Idesign tool to compensate for assumptions.
 - Icing engine model assuming ice build up effects LPC/HPC performance maps and ice ingestion reduces energy in the LPC and HPC

Engine Geometry:

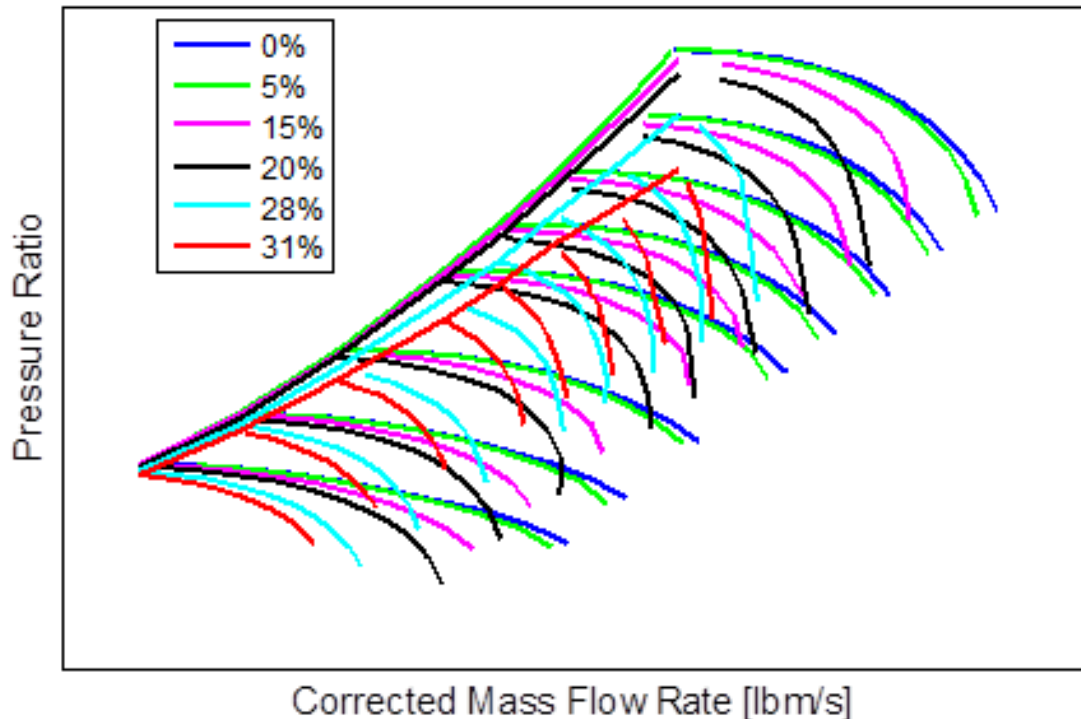
- 2 Spool
- High Bypass
- HPC bleed cooling flow to the HPC and LPC



Engine Modeling during Icing event:

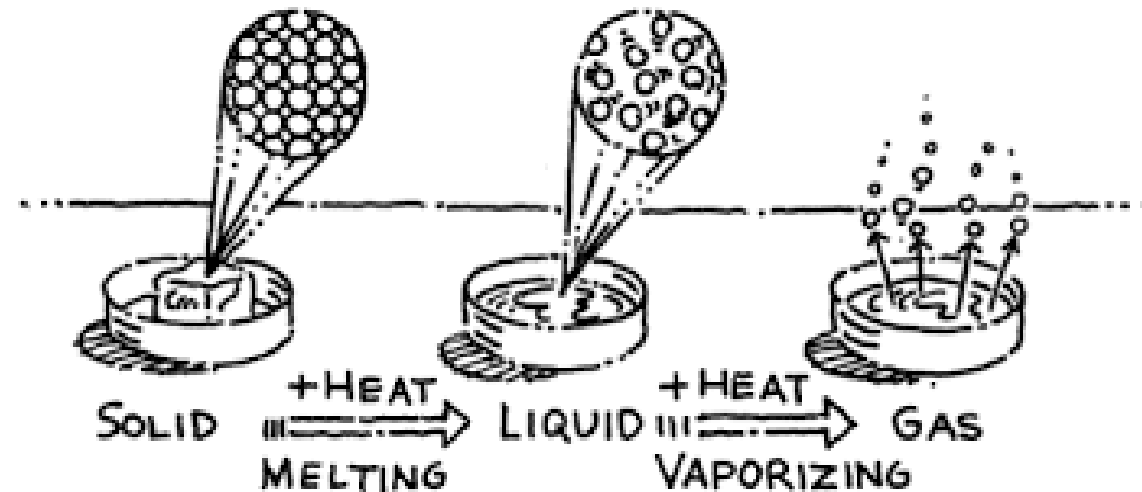
LPC performance effect:

- As ice builds up the LPC becomes blocked resulting in loss of pressure ratio and mass flow at a given shaft speed



Ice ingestion effect

- Non-Adiabatic process. Energy (Q) is lost as ice in the air stream melts then vaporizes



Source: thinknut.blogspot.com

Diagram of icing simulation

High bypass two spool engine

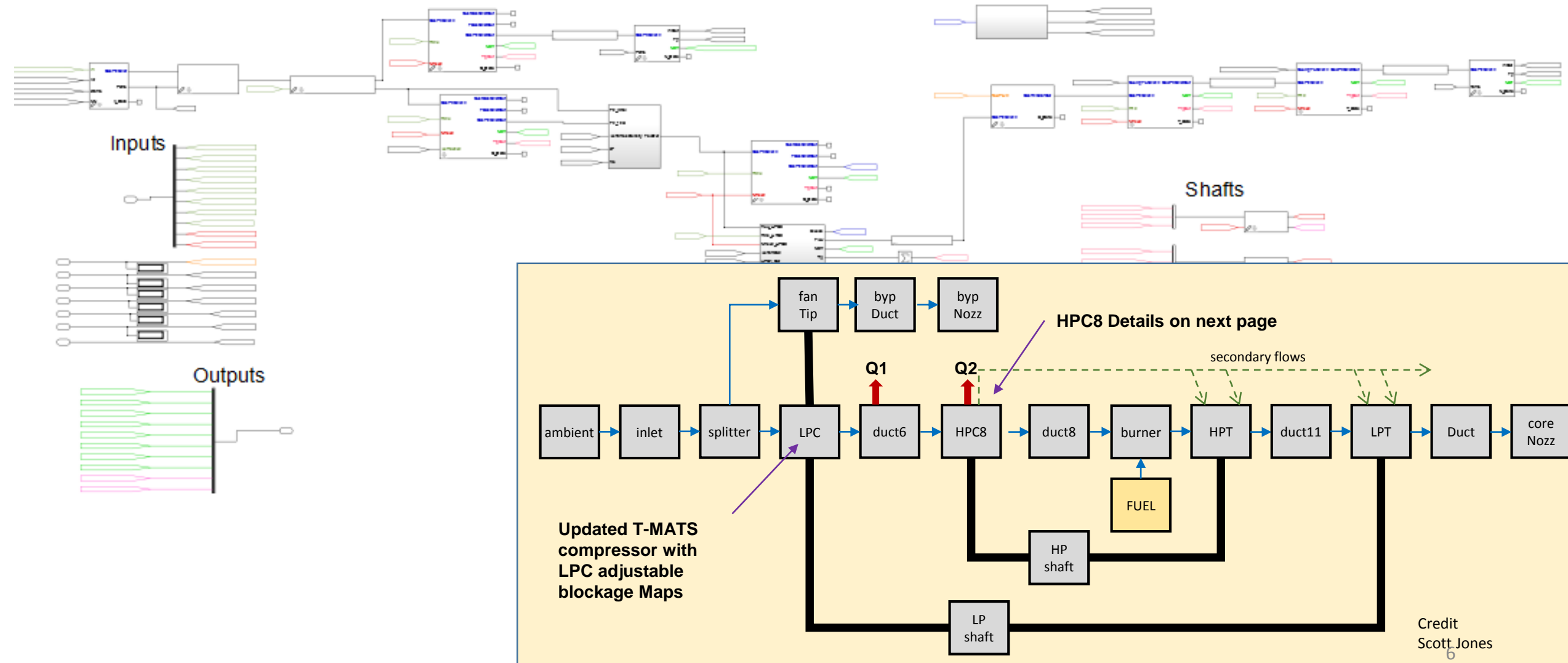
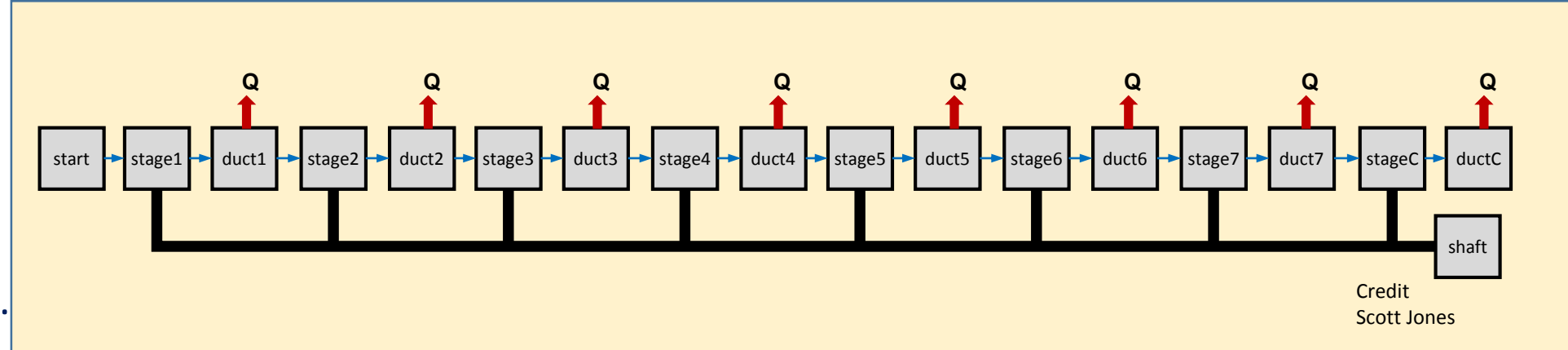
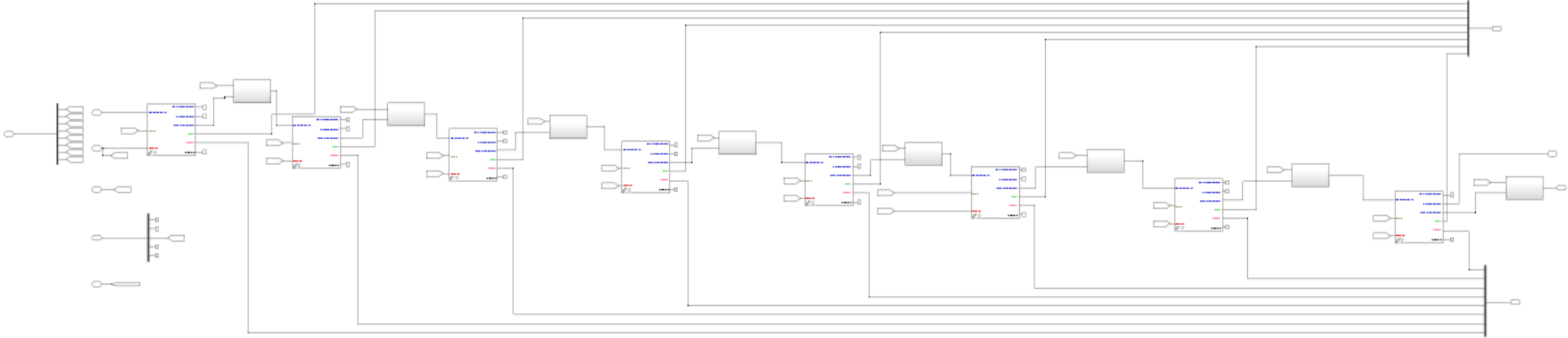


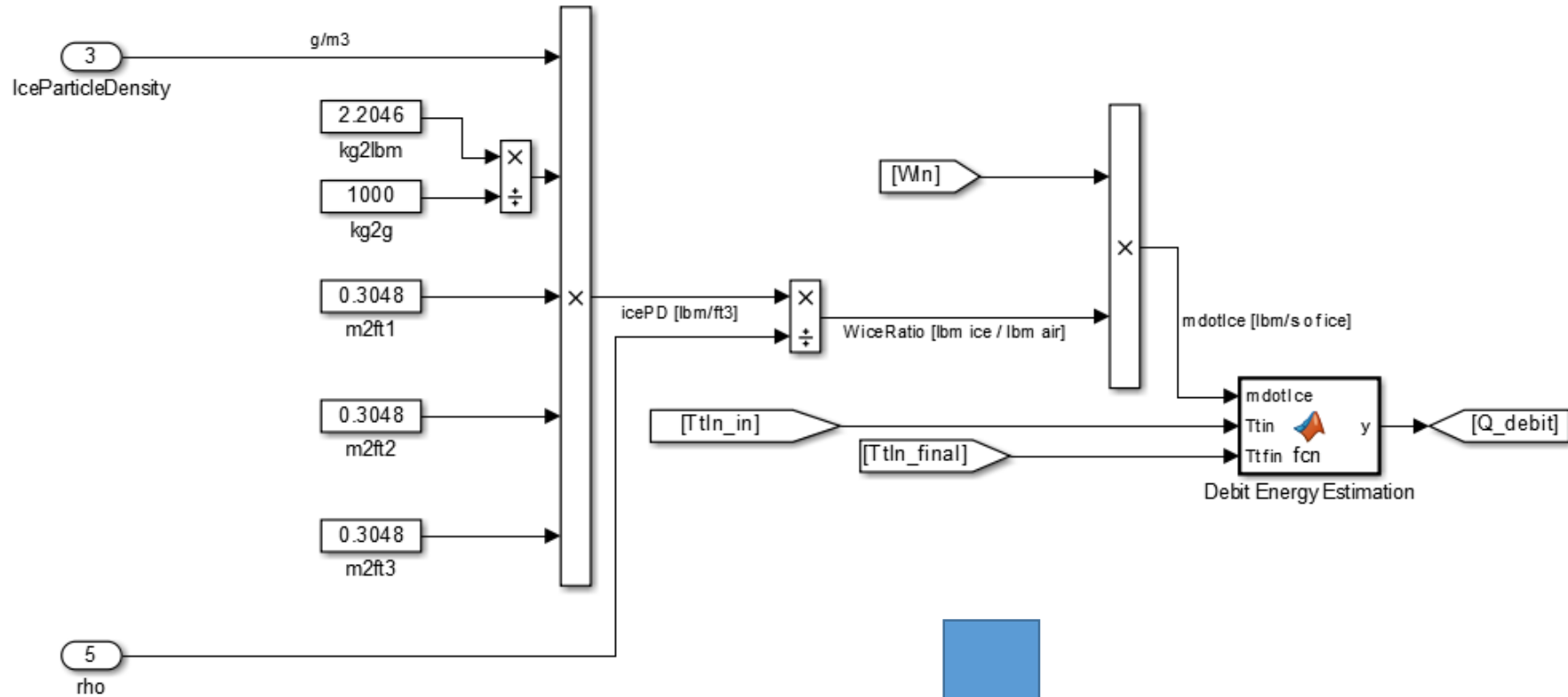
Diagram of 8 stage compressor



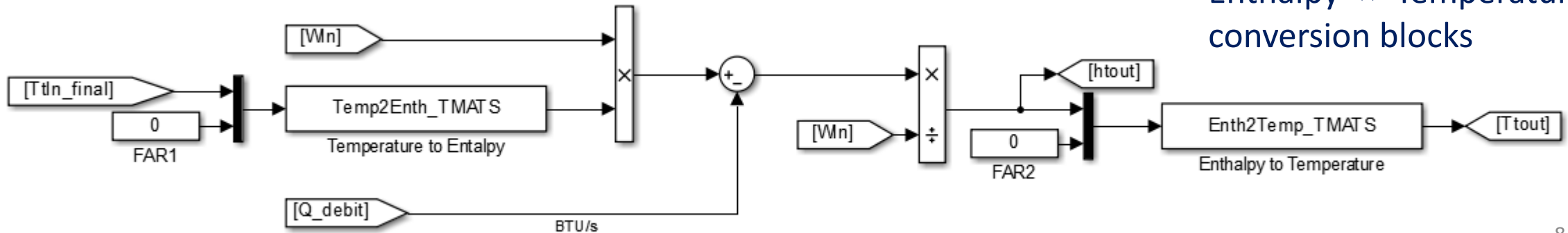
To better approximate
non-adiabatic compression.

The HPC is simulated stage by stage (8 separate
stages) to allow element phase change energy to
be removed mid compressor.

Component calculation of Q



Q debit calculated based on temperatures before and after a component as well as the phase specific heat, heat of fusion, and heat of vaporization of H₂O



Q removed from flow as enthalpy using the T-MATS Enthalpy ↔ Temperature conversion blocks

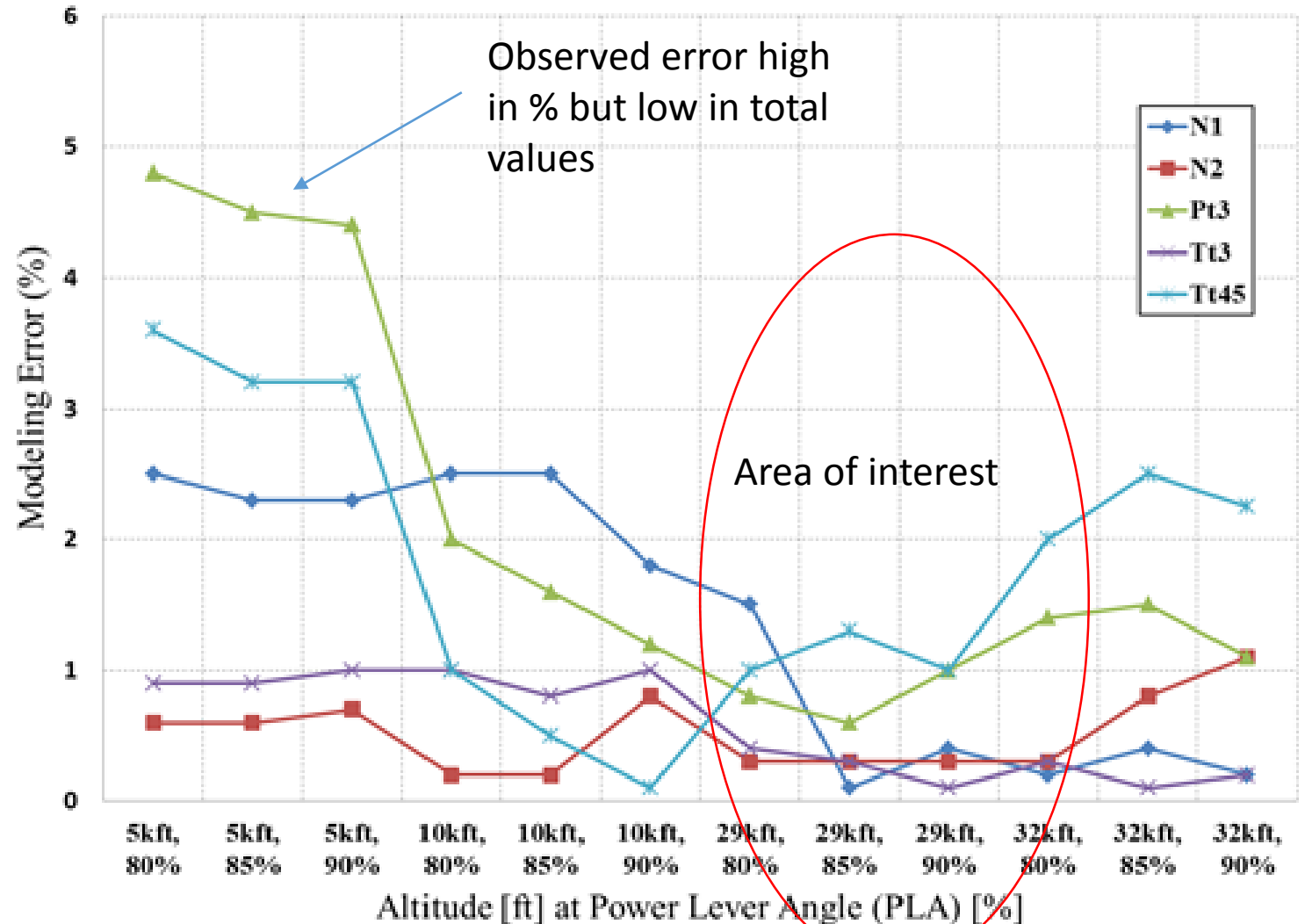
Simulation vs Baseline SS Engine Data:

Assumptions:

- HPC fractional bleed values set to % of HPC flow.
- All stability bleed valves were assumed closed

Results:

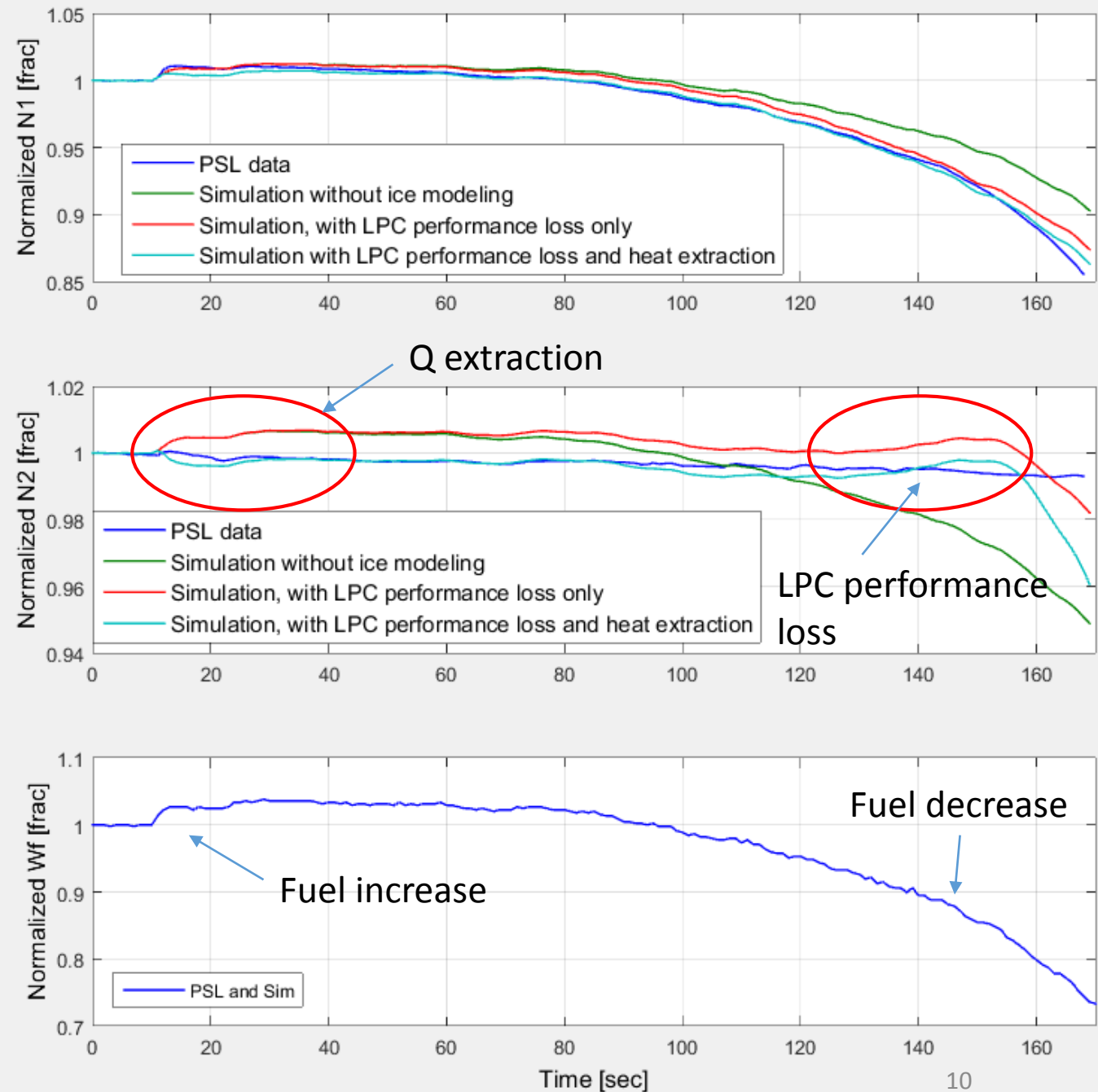
- Errors below 1% for the majority of parameters in area of interest.
- Results comparable to previous modeling work.



Dynamic simulation of icing event.

Simulation

- Fuel flow values from PSL data are fed into the T-MATS simulation
- Ice cloud injected at 10sec.
- Initially, Engine requires more fuel to melt ice cloud (Q extraction)
- Ice accumulation partially blocks the LPC reducing performance. Because less work is being done by the LPC, N2 speed can be maintained as fuel is reduced. (LPC performance loss)



Conclusion & Future Work

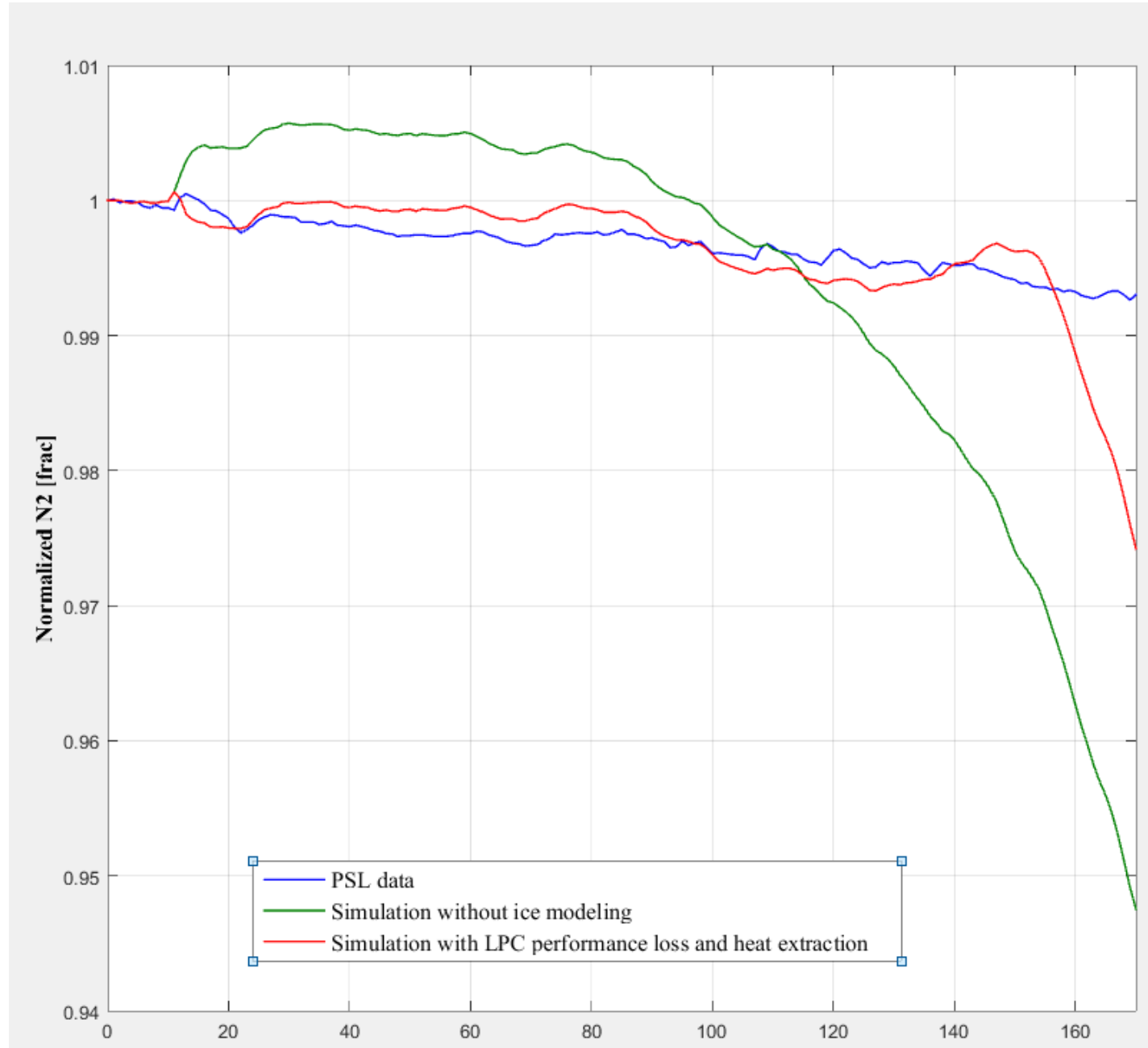
A T-MATS engine model was created to test an ice buildup detection algorithm.

- Baseline testing showed good matching with baseline steady state engine data.
- Ice cloud ingestion modeling took two forms:
 - Heat extraction due to melting ice
 - LPC pressure ratio and mass flow drop with ice buildup on turbomachinery components
- Dynamic simulation of icing event shows good matching in the area of interest.

Future work planned to begin again next year:

- Developing and testing a realistic controller for the engine
- Implementing an ice detection algorithm that identifies the icing event within an allotted timeframe.
- Long Term assuming continued funding and project success:
 - Test icing detection algorithm in the PSL

Backup



Model Extraction by System Identification and Advanced Control Design in T-MATS

Hanz Richter

Control, Robotics and Mechatronics Lab
Cleveland State University

Xian Du

School of Aeroengines and Power
Northwest Polytechnic University, China

Overview

1. System Identification (SI)
2. SI as an alternative to perturbation for linearization. Initial results with JTD9 / T-MATS
3. Model-based control design: State-Space Sliding Mode Control (SMC)
4. LPS speed control with SMC. Initial results with JTD9 / TMATS
5. Conclusions
6. Suggested work

System Identification

- SI is a model-fitting technique based on input/output observations of a process
- Radically different from Jacobian linearization from perturbations
- A model structure needs to be supplied (transfer function, state-space, etc.)
- Once a “blank” model is defined, optimization is used to find model parameters.

Identification Parameters

SI can use experimental data or simulated data. In our case, it's a numerical experiment. Some prior knowledge about the system is needed to determine the best "experiment parameters":

- The user must define an excitation signal, typically a swept-sinewave (chirp). Parameters: Amplitude, bias, initial and final frequencies, sweeping time, linear vs. log sweep.
- I/O data must be sampled at a certain interval T_s . This is chosen based on expected model bandwidth (T_s too large: aliasing; too small: excessive data, noise emphasized)

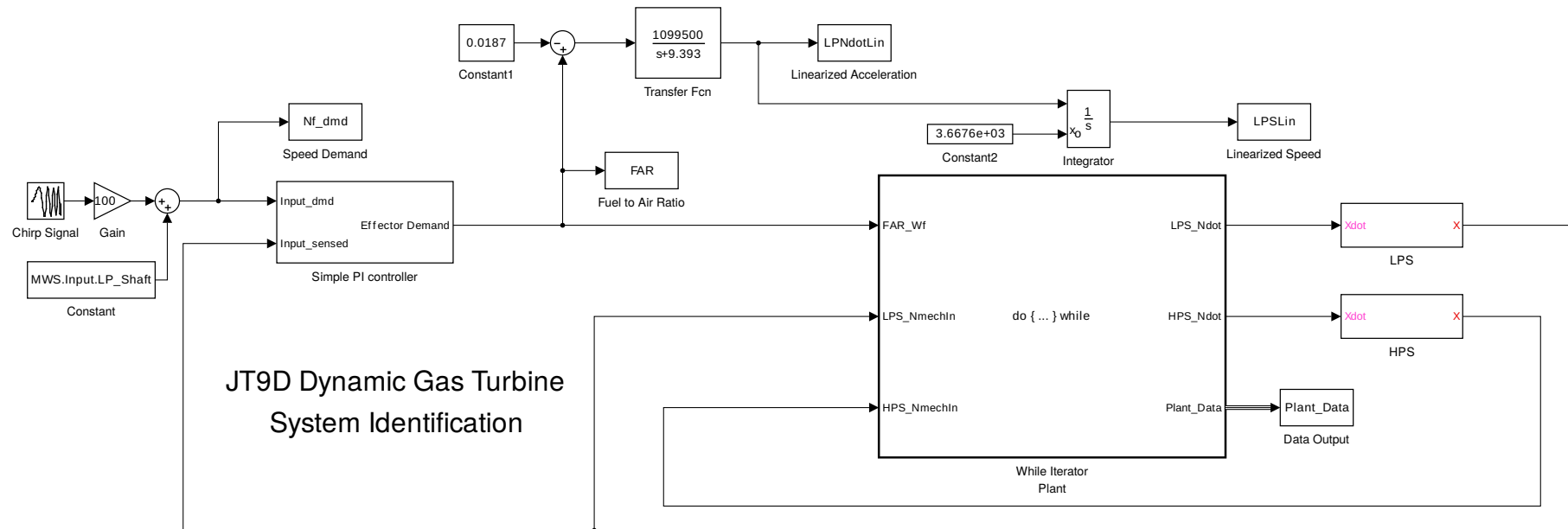
The amplitude and bias must be chosen to keep the model responding in the linear regime. In models with nonlinear DC gains, these must be customized according to the linearization point.

Running the Identification

Fortunately, software tools are available to carry out these tasks very efficiently (Matlab System Identification Toolbox), open-source: Scilab, Octave.

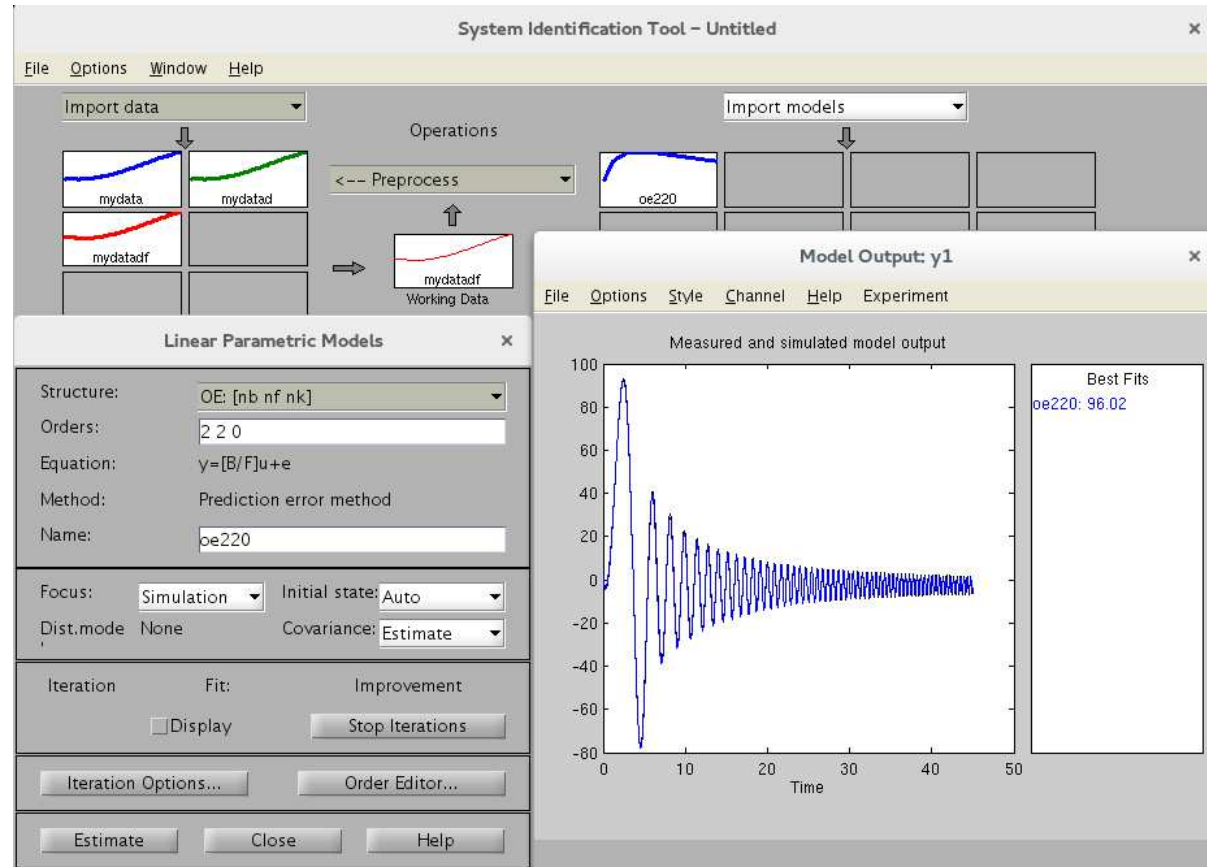
1. The model to be identified can be in open-loop or closed-loop. In any case, plant input is recorded.
2. The dataset is imported and pre-processed (mean removal, filtering)
3. In engine linearization mean removal is fundamental, to capture only the Δ variables.
4. A fit score is obtained for each trial model structure.

Linearizing the JTD9 Model



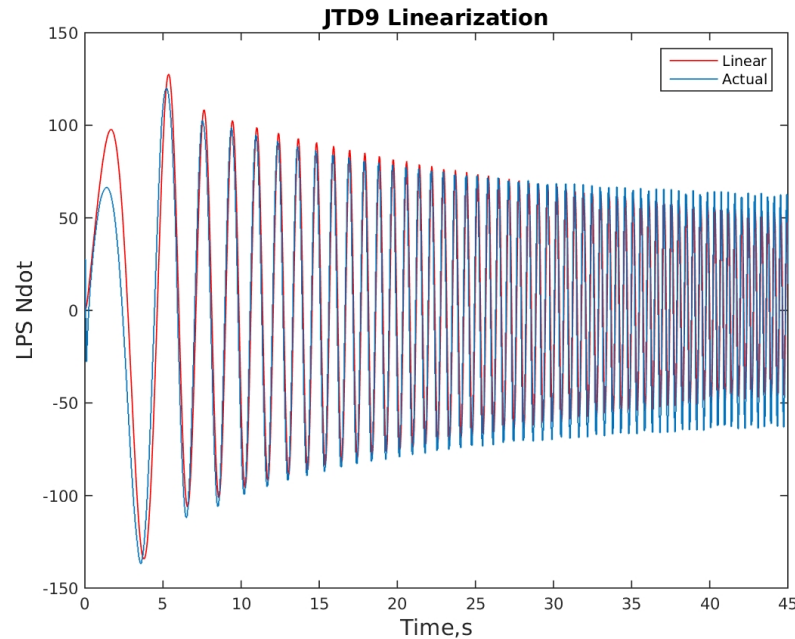
- Chirp: ± 100 rpm from steady LPS speed, from 0.1 to 2.5 Hz in 45 sec, linear sweep.
- Record FAR and LPS speed, sampled at 0.04 sec (same as simulation)
- 2nd-order dynamics expected (one pole from FAR to LPS accel, then integration to LPS speed).

Model Fit in Sys Id Toolbox



- Removed means (trim offsets), low-pass filtered at 5 Hz (note sampling is at $1/0.04 = 25$ Hz).
- Chose transfer function structure with 2 poles, 2 zeroes, no noise channel
- Chose output error minimization. Resulted in a 96 score (excellent)

Fitted Model in T-MATS



The model is first identified in discrete-time. After conversion to continuous-time (zero-order hold equivalent) and removal of high-frequency zeroes (artifacts):

$$\frac{\Delta N_{LP}(s)}{\Delta FAR(s)} = \frac{k}{(s + 9.393)(s + 0.4967)}$$

Note: The pole at -0.4967 is found by SI as an approximation to the integrator. We can manually force an integration if needed.

Using the Model for Controller Design

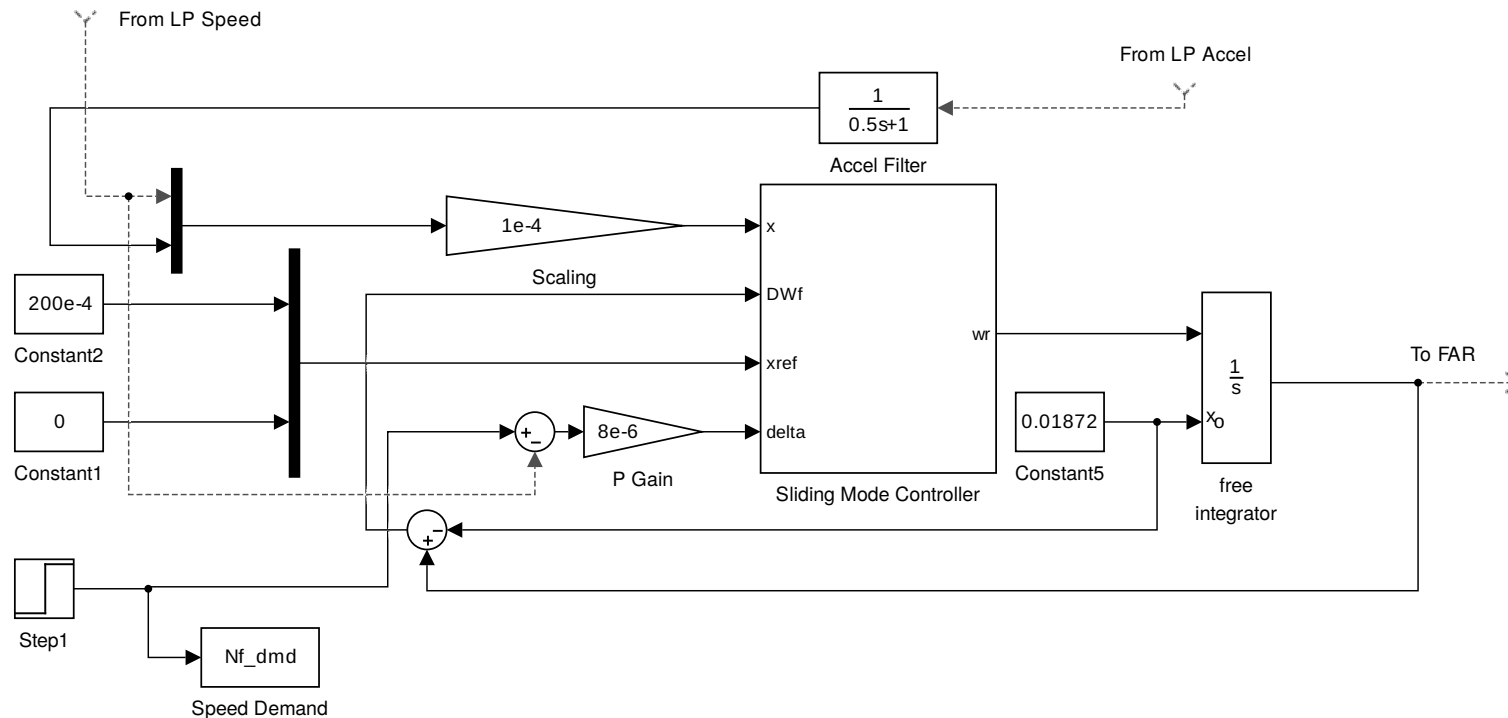
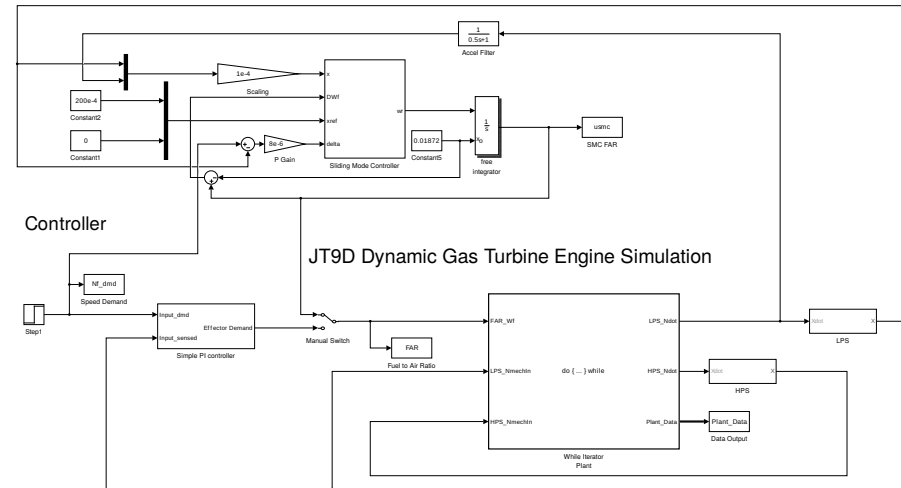
Due to units (rpm and FAR scale), k is very large. It's better to scale the model. A scaling factor of 1×10^{-4} was found to be adequate. With this, the model becomes

$$\frac{\Delta N_{LP}(s)}{\Delta FAR(s)} = \frac{109.95}{(s + 9.393)(s + 0.4967)}$$

As an example, a state-space sliding mode controller was designed using this model.

- SMC is known for its robustness (can work well without re-tuning over a large portion of operating envelope)
- SMC has strong disturbance rejection properties
- SMC is simple, few and intuitive tuning “knobs”

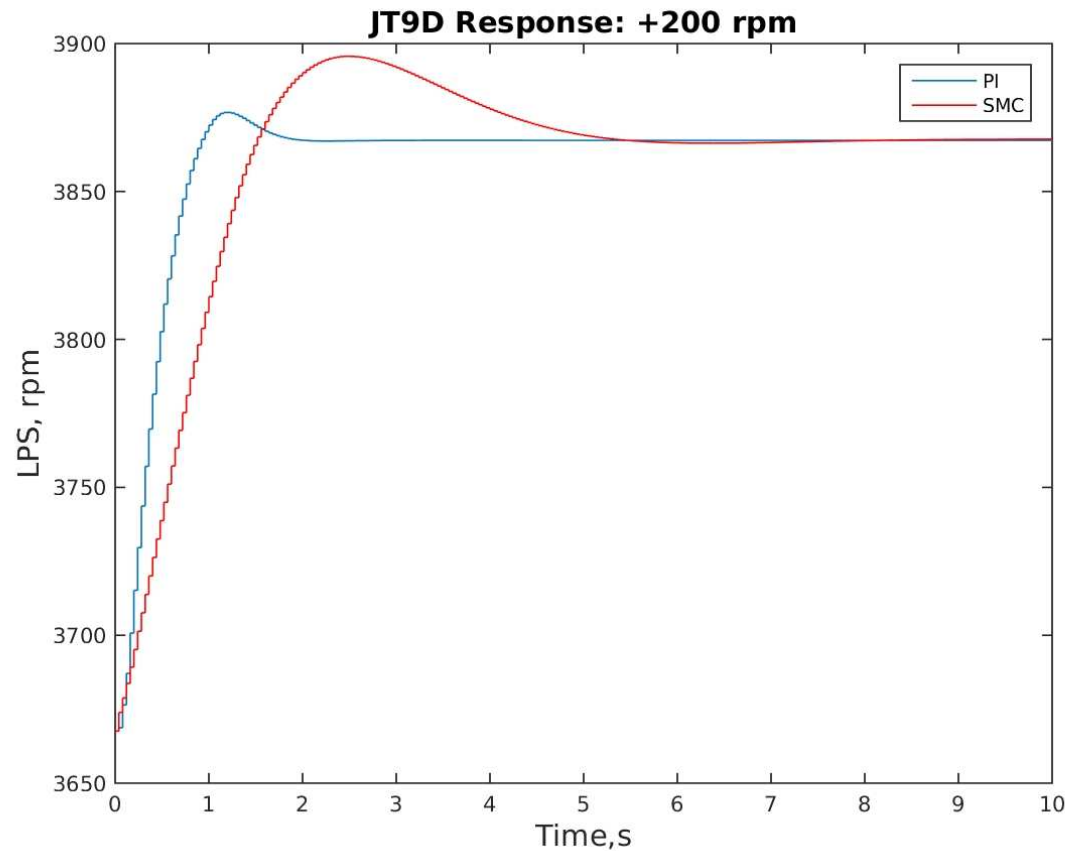
SMC Implementation in T-MATS



State-Space SMC Drawback

- Input integration was used in this model (augmented state vector: speed, acceleration, Δ FAR). Target values must be provided!
- Speed and acceleration targets: desired rpm change and zero, resp. (OK)
- Δ FAR target? -The controller needs steady map info in advance!
- Quick fix: calculate Δ FAR target from speed target and linearized model (this will be inaccurate). Add a self-correction P-loop for Δ FAR target.
- This has worked well (no steady-state error).
- This implementation uses LPS speed and acceleration as states. Need a lowpass filter to break algebraic loop and enable accel. feedback.

Results: +200 rpm



Smaller overshoot, faster response. Response qualities will be more uniform across envelope in comparison with PI.

Conclusions

1. System identification is a straightforward alternative to perturbation-based, Jacobian linearization in T-MATS.
2. Extracted models can be used for many advanced controller designs, for instance SMC.
3. T-MATS makes controller evaluation with nonlinear engine models fast and convenient (thanks to the developers!)
4. Open-source approach solves limitations with C-MAPSS access by foreign researchers

Desirable Features

Some advanced engine control systems are multi-input (active controls on bleeds, vane angles, etc). These extra degrees of freedom can improve transient response, reduce fuel consumption and handle engine limits.

Preliminary work (Xian Du):

- Use fuel flow and stator vane angle to simultaneously control fan speed and HPT temperature: MIMO SMC
- Extend min-max logic to add limit protection
- Limit controllers switched on upon approaching limit
- Main regulator resumes tasks after preset dwell time

Results

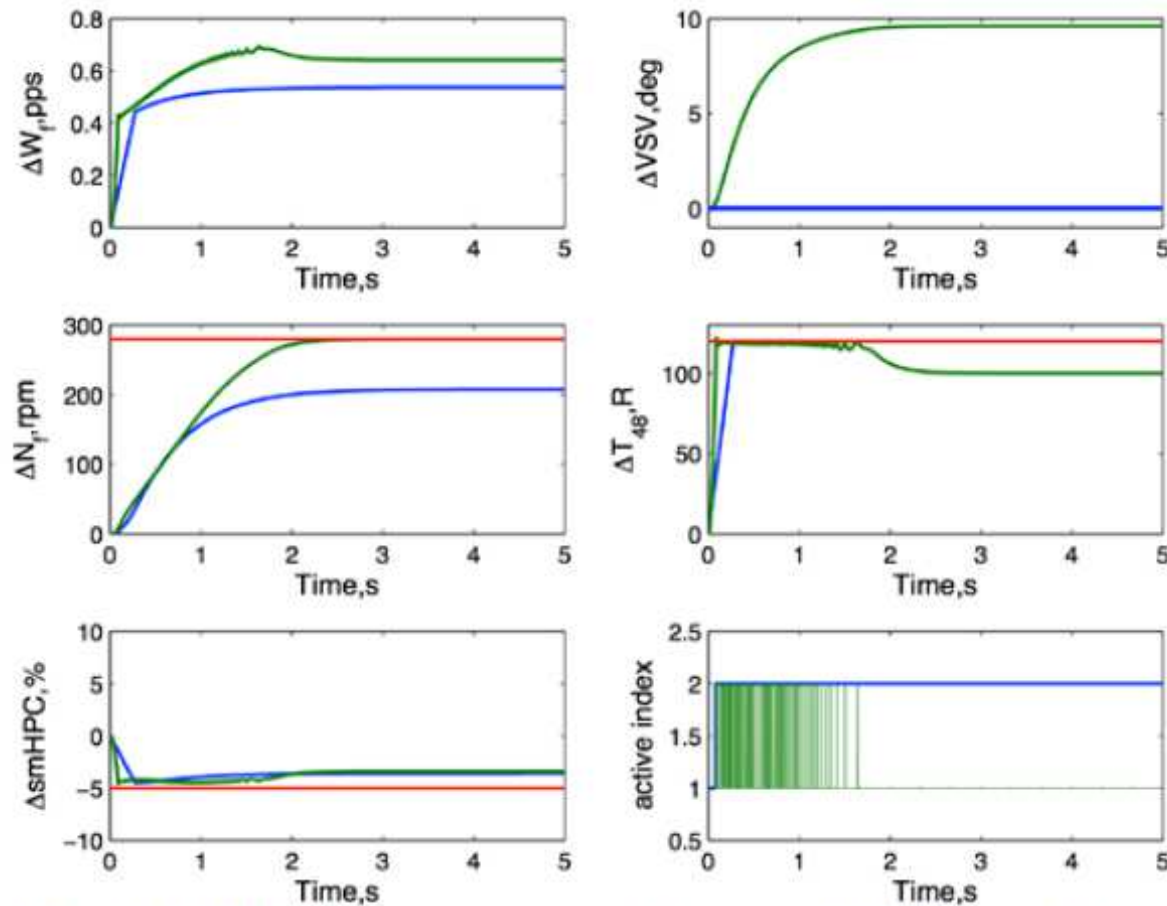


Fig.5: MIMO Simulation (green line) vs. SISM Simulation (blue line) for Target Tracking Study

Paper submitted to 2015 ASME Dynamic Systems and Control Conference, Columbus, OH.

Suggested Work

1. Perform SI using FAR as input but LPS and HPS as outputs. Will result in 2 transfer functions with the same poles. This will replace acceleration feedback with HPS speed feedback, much better.
2. Perform a comprehensive SI process across the operating envelope, including critical outputs (temperatures, pressure ratios)
3. Test single-input min-max limit protection logic and controllers using these models.

Using T-MATS for Supporting the Design of Bleed System Controls

Umer Khan

*The Toolbox for the Modeling and Analysis of
Thermodynamic Systems (T-MATS) Workshop*

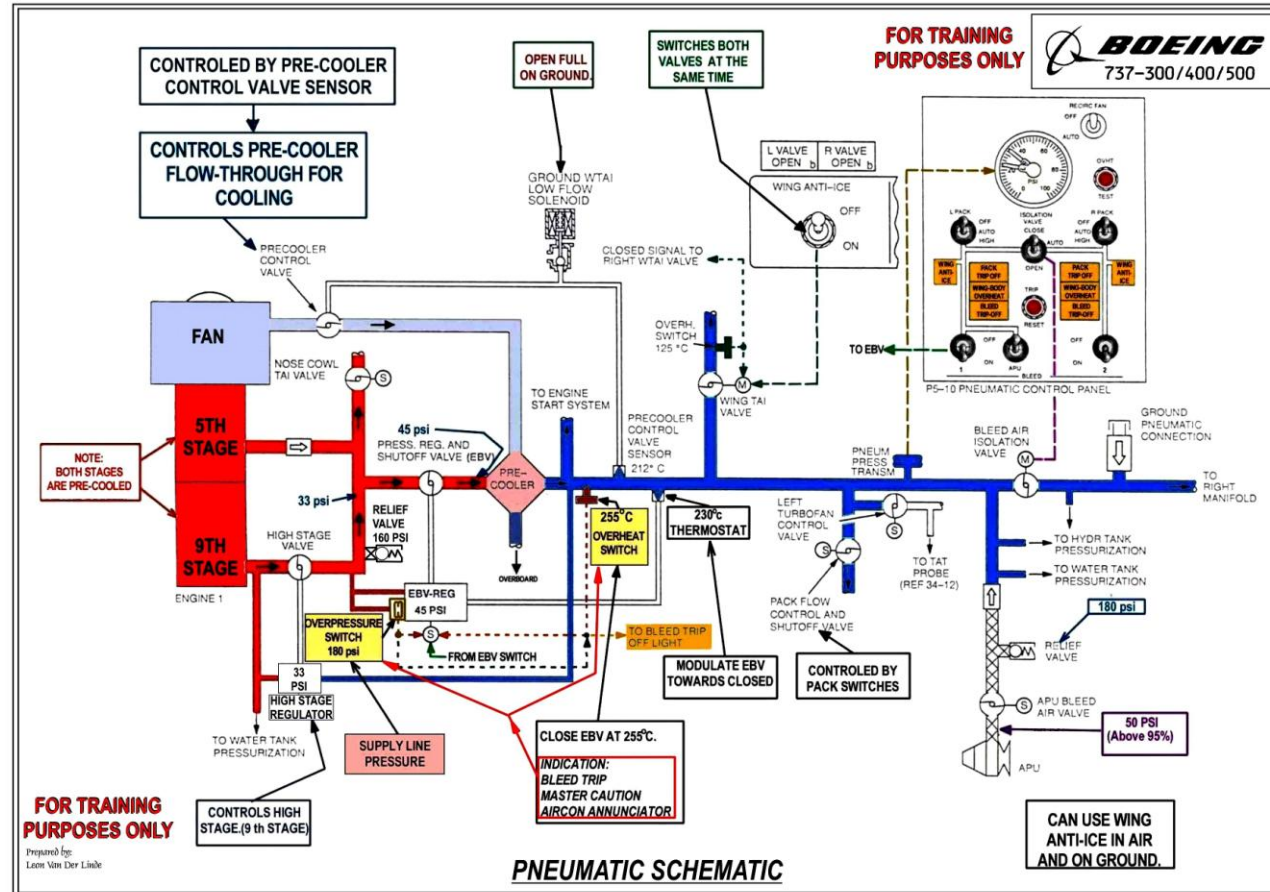
April 15, 2015

Outline

- Introduction
 - Aircraft Bleed Air System
 - Motivation
- Model Requirements
- Approach
- Model Tuning
 - Steady state and dynamic
- Model Capabilities
- Conclusions and Recommendations

Aircraft Bleed Air System

- Bleed air used for:
 - Cabin pressurization
 - Air conditioning
 - Engine start
 - Aircraft anti-ice systems
- Bleed air extracted from compressors of engine
 - P and T varies during flight



Motivation

- Previously used extremes of flight envelope
- Advancement of aircraft and engine technology and use of composite materials in aircraft
 - Operation at higher pressure and temperature
- Need the ability to model bleed flow variations through flight envelope

Requirements

- Engine model: dual-shaft turbofan engine
 - Capable of simulating operating conditions based on N1, P, T and Mach
 - Tunable to match OEM data
 - Simulink environment in real time
 - Produce transient engine bleed outputs (P and T)
- Evaluated several alternatives, T-MATS deemed most efficient for this purpose
 - JT9D model
 - iDesign

Approach

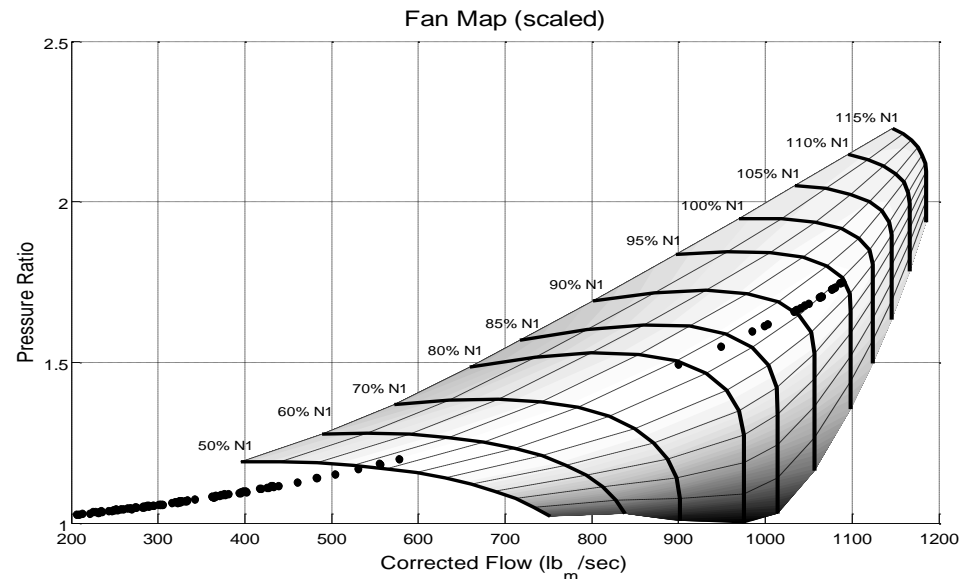
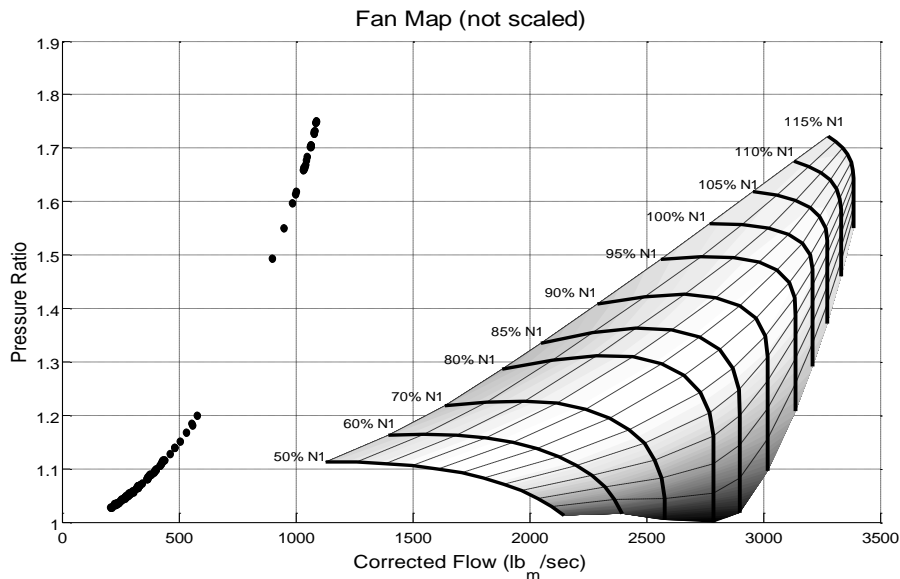
- OEM data
- Scale maps using iDesign feature
 - Compressor maps
 - Turbine maps
- Test/tune model against known data
 - Steady state
 - Transient

OEM Engine Data

- One design point identified
- Engine data limited to fan and some compressor parameters
 - Extracted data required for scaling using thermodynamic laws
- Turbine operating point assumed based on compressor operating point

T-MATS iDesign

- Used iDesign feature along with chosen design point to find map scaling parameters
- Operation to idle required map extrapolation



Steady State Performance

- Compared model performance with OEM data

- Fan Bleed Delta

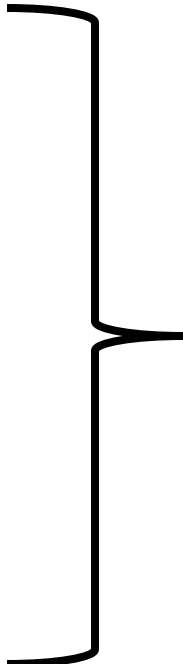
- P: 1.7 % (0.2 Δ psia)
 - T: 0.3 % (1.6 Δ degF)

- IP Bleed Delta

- P: 6.2 % (2.7 Δ psia)
 - T: 3.1 % (23 Δ degF)

- HP Bleed Delta

- P: 20 % (21 Δ psia)
 - T: 13 % (148 Δ degF)

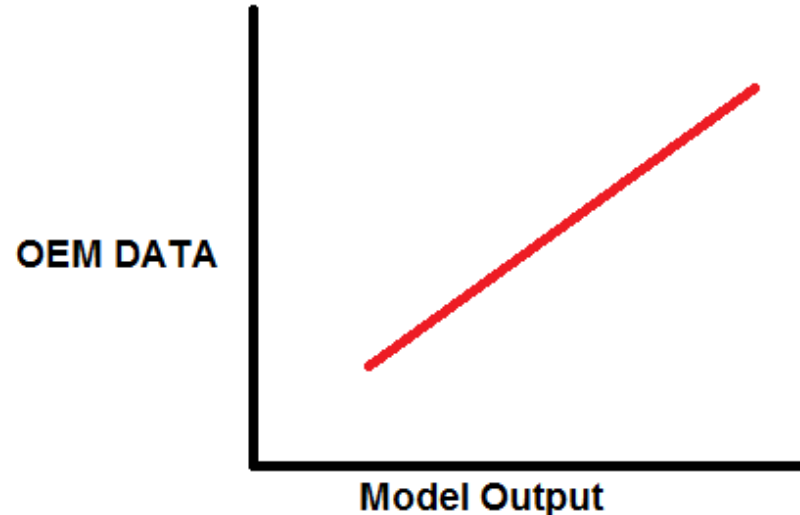


Can be reduced by post-processing calibration

Complete fan data from OEM gave most accurate results

Post Processing Calibration

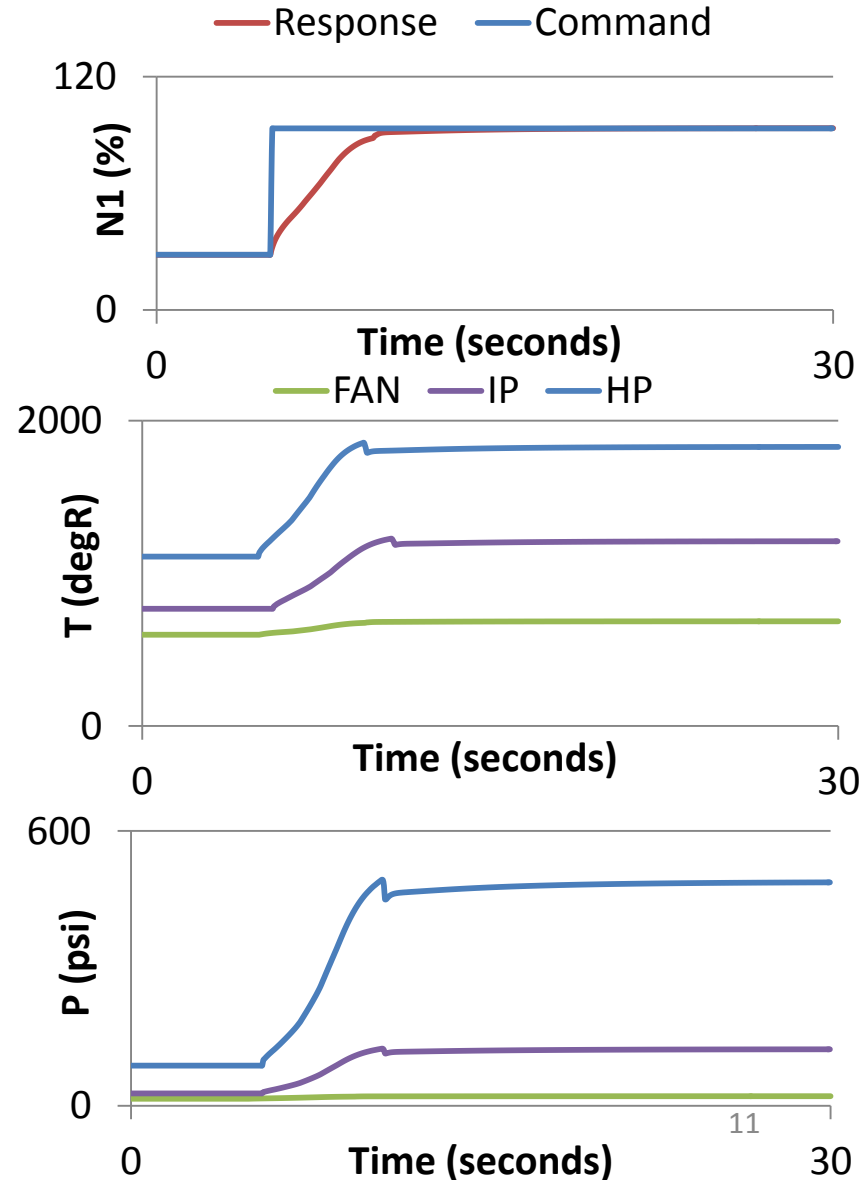
- Linear relationships between model output and OEM SS data
- Using these relationships:
 - Fan Bleed Delta
 - P: 1.7 % → **1.3 %**
 - T: 0.3 % → **0.3 %**
 - IP Bleed Delta
 - P: 6.2 % → **3.1 %**
 - T: 3.1 % → **1.3 %**
 - HP Bleed Delta
 - P: 20 % → **7.8 %**
 - T: 13 % → **1.6 %**



Linear relationship implies T-MATS model captures characteristics of real engine SS behavior

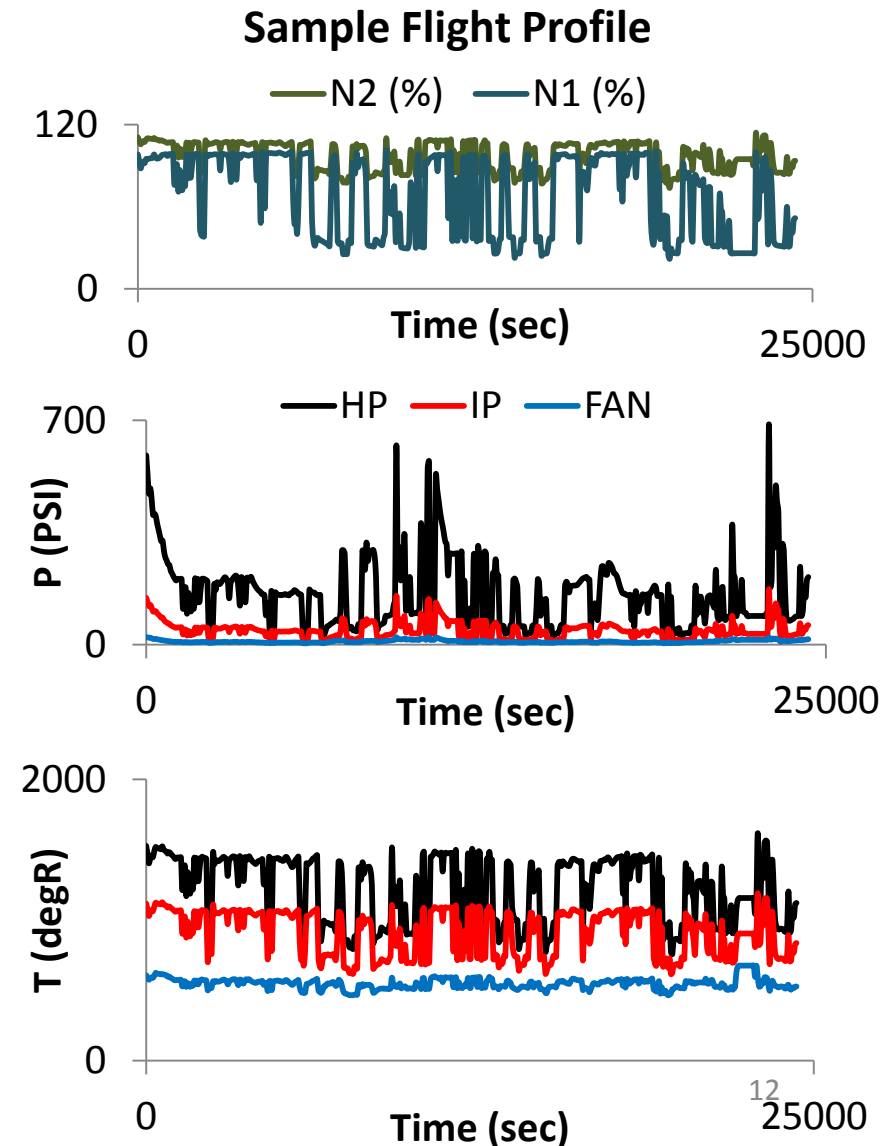
Dynamic Performance

- OEM transient data
 - N1 step command
- PI tuned using Simulink Response/Parameter Optimization
- Gain scheduling based on flight altitude and mach



Model Capabilities

- Simulate engine operation within flight envelope
 - Idle to maximum throttle
- Ability to see fluctuations in parameters during flight profile:
 - Bleed Data (P and T)
 - Engine and Core Flow
 - N2 (%)



Conclusions

- Functional dual-spool engine model with required outputs
- Ability to execute model dynamically over engine operating range
- Delta tolerances, with post-processing, currently workable

OVERALL EXPERIENCE WITH T-MATS VERY POSITIVE

Experience with T-MATS

- Positive
 - Duel-spool engine model example available
 - iDesign
 - Multi-variable iterative solver
 - Simplicity in model layout
- Suggestions
 - More documentation required to assist users (include publications)
 - Strategies on using iDesign with limited data, which assumptions can be used
 - Generic engine maps for different classes of engines
 - Rapid accelerator mode, RTW code generation (*currently being resolved*)
 - Include simplified generic FADEC controls in T-MATS model library

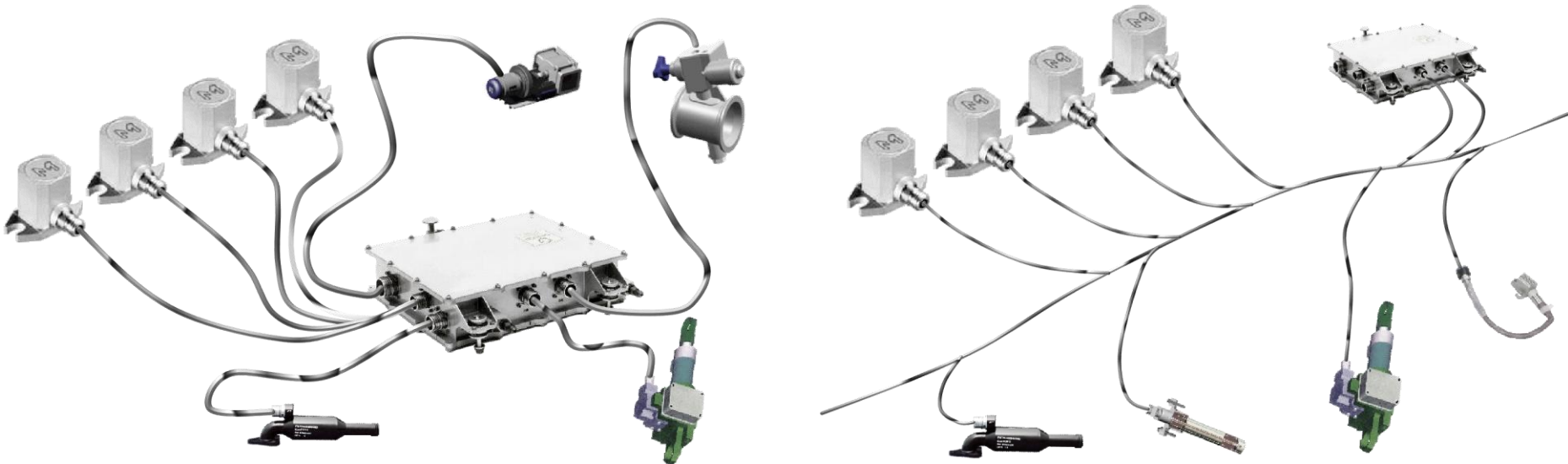
THANK YOU

Advanced Propulsion Control Technologies

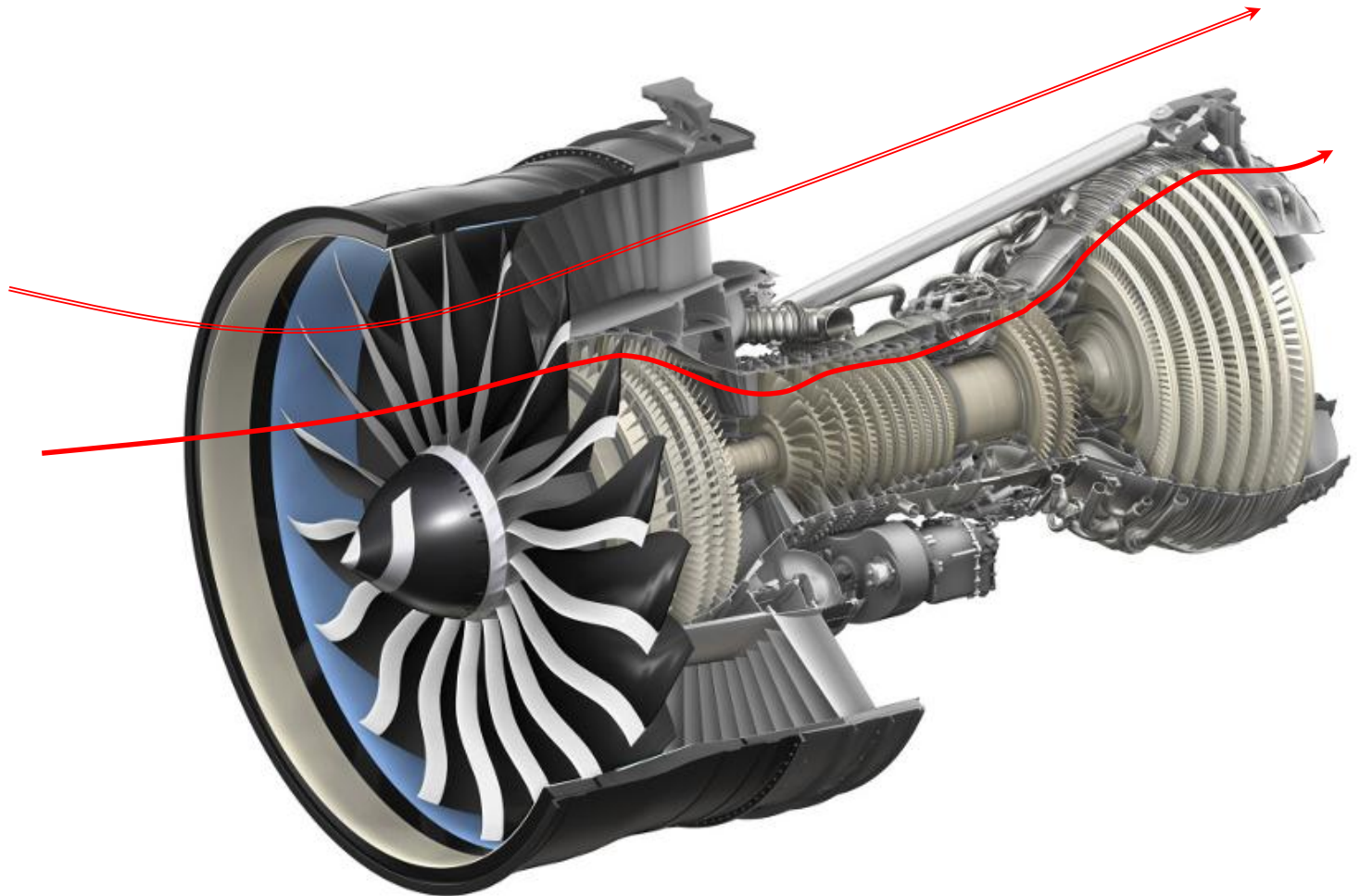
Dennis Culley

TMATS Workshop
NASA Glenn Research Center

April 15, 2015



Much of what we think about in controls has to do with gas path dynamics and optimizing the overall operation of the turbine engine ...



... but controls is a technology that exists at the system level, at the local level, and as a separate engine system.

The purpose of controls is to safely navigate from one steady state condition to another and, for a given condition, to maintain the machine at optimum performance



The Global Focus of Control System Technology

Distributed Engine Control

- Separating the control law processing function allows it to be located in a smaller volume in a more accessible and less hostile environment.
- This leads to more design choices for thermal management and other system benefits.
- Its modularity leads to a simplified growth path for access to mainstream microprocessor technology and more computational capacity.

Model Based Engine Control

- More computational capacity leads to more complex control law and improved system performance.

The Local Focus of Control Technology

Distributed Engine Control

- Immediate benefits include trimming out variation in sensors and actuators for more precise control.
- Local processing allows higher bandwidth signals and the potential for extracting more information from a limited suite of control elements.
- Locally, control technology improves the quality of data available to perform control.
- Embedded control in smart nodes provides a growth path for advanced control technology.

Advanced Combustion Control

- Modulation of fuel flow to eliminate combustor instability in lean burning combustion processes.

Controls as an Engine Hardware Subsystem

Distributed Engine Control

- Its modularity defines partitions that simplify control integration.
- These partitions ease the impact of obsolescence, namely cost and system availability.
- Digital communications between modules reduce interface complexity while lowering system harness weight and improving reliability.
- The flexibility of distributed system hardware partitioning also helps reduce constraints between the engine mechanical/thermal and electrical/electronic systems.

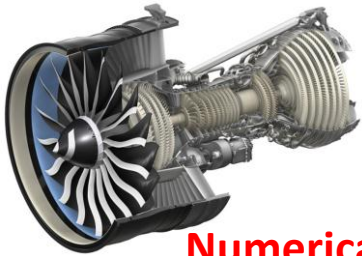
Control technologies cannot advance system performance unless it takes an active role in the system design. Absent this interaction, controls can only optimize for a given design.

TMATS Workshop

Is control system research reactive or is it an active participant in propulsion system design?

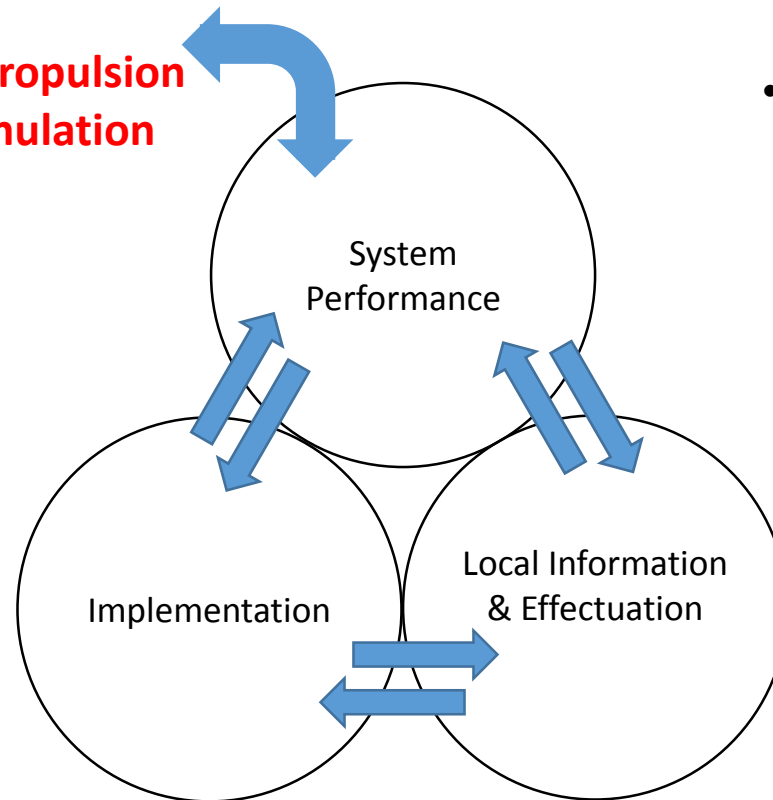
Which control technologies provide the most system benefit relative to the investment in their development?

Control Technology Development



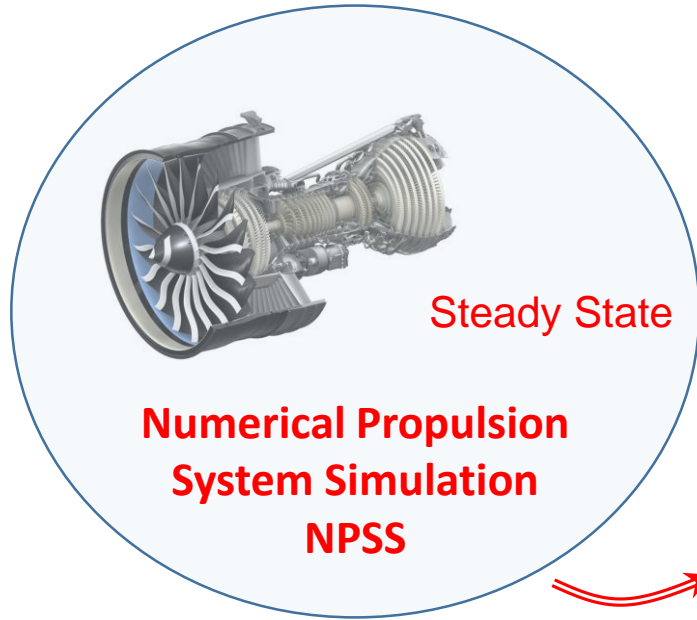
Numerical Propulsion System Simulation

- Creating a more flexible and reliable hardware environment.
- Creating a growth path for future technology.
- Maintaining access to cutting edge electronics.

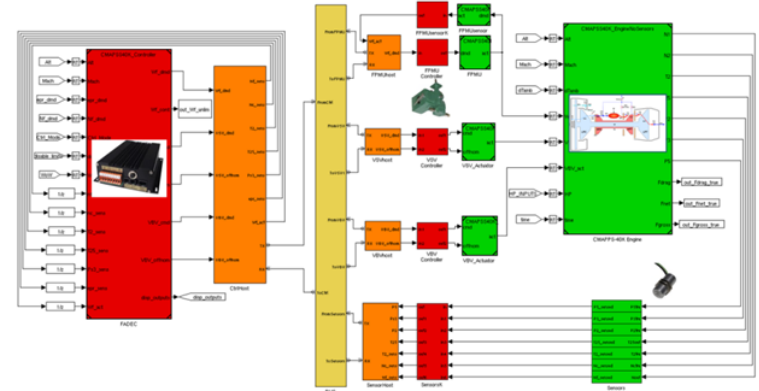


- Creating and extracting more information from the existing data.
- Using real-time on-board models to improve engine performance
- Improving the accuracy and responsiveness of sensors and actuators.
- Creating more system information without adversely impacting cost and reliability.
- Providing a more adaptable engine

Tool Development Needs



Dynamic

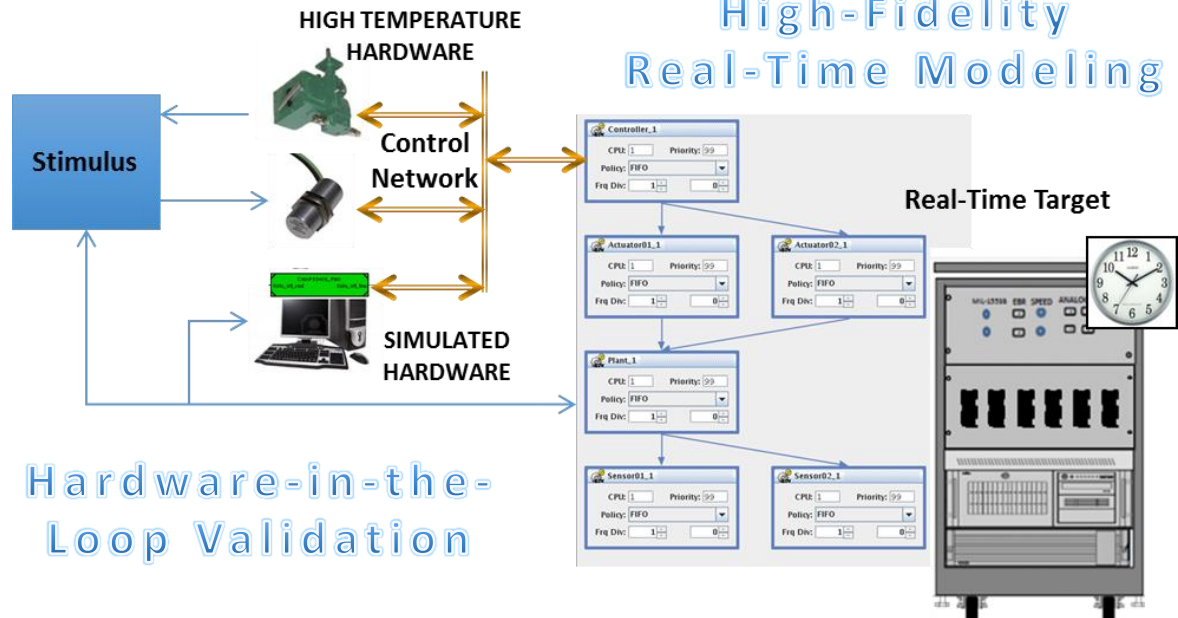


**Toolbox for the Modeling
and Simulation of
Thermodynamic Systems
T-MATS**

**Control Concept
Development**

**Evaluation of
Impact on
Engine Design**

**High-Fidelity
Real-Time Modeling**



**Hardware-in-the-
Loop Validation**

The Role of T-MATS

Control Concept Modeling

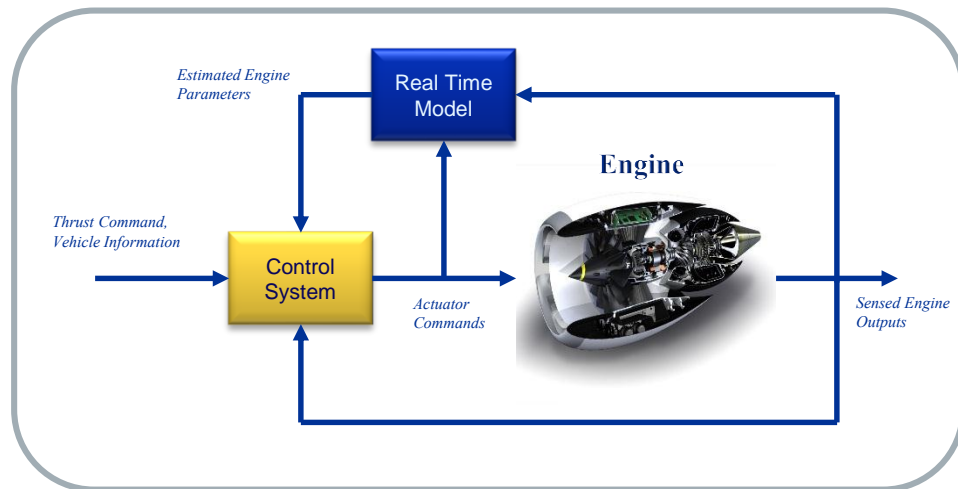
- Porting the NPSS model into the controls development environment can be very labor intensive.
- The user should have full control over the trade-off between model fidelity and speed.
- Full access to the entire Engine Model is required to evaluate new control concepts, including actuation to enhance participation in the engine design process.

Real Time Hardware-in-the-Loop Validation

- All control system development must eventually be evaluated in hardware. This requires developing real time code for the HIL target computational platform.

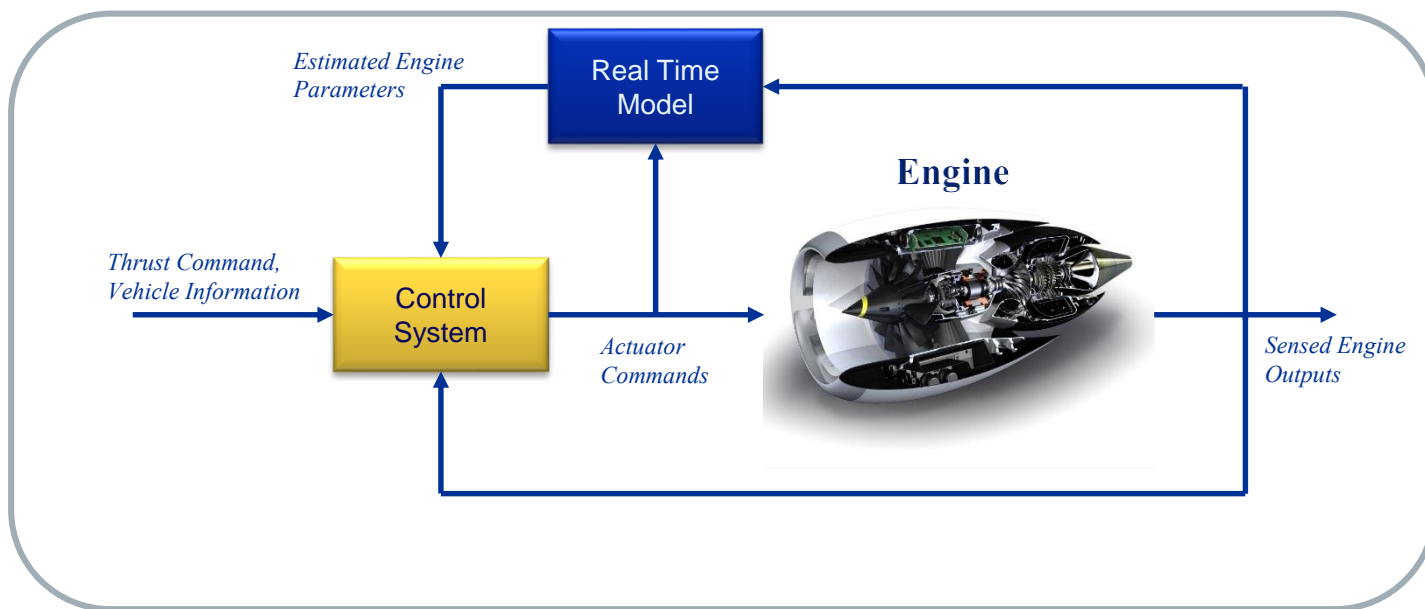
Model-Based Engine Control using T-MATS

Jeffrey Csank and Joseph Connolly
Intelligent Control and Autonomy Branch
NASA Glenn Research Center



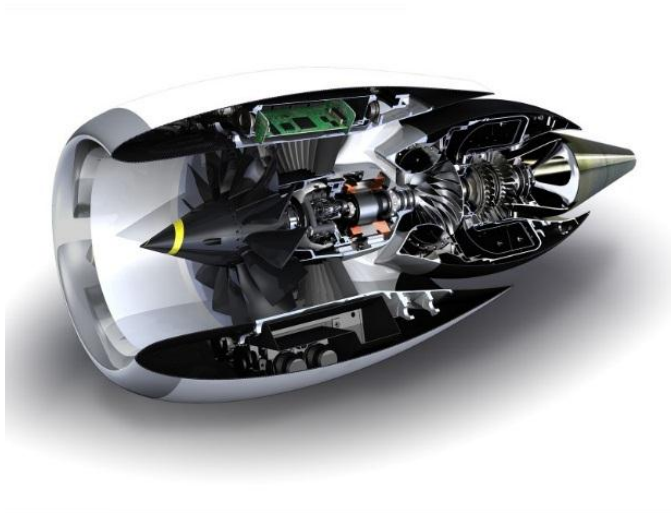
Project Goals

- Implement Model Based Engine Control architecture and test on turbofan engine
- Steps to hardware engine test with MBEC:
 - Develop high fidelity transient engine model
 - Develop MBEC architecture
 - Implement MBEC in hardware and demonstrate on test bench
 - Hardware test



Actual Engine

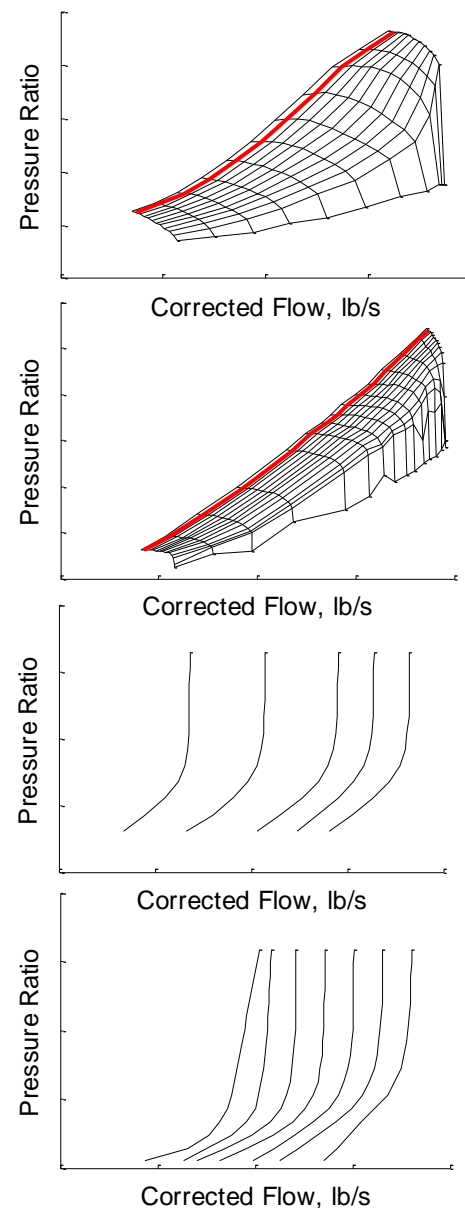
- Flight envelope: $<25,000\text{ft}$, $<0.35\text{ Mach}$
- High Bypass Turbofan (~ 8)
- Geared Fan ($\sim 3:1$)
- $\sim 600\text{ lb Thrust}$





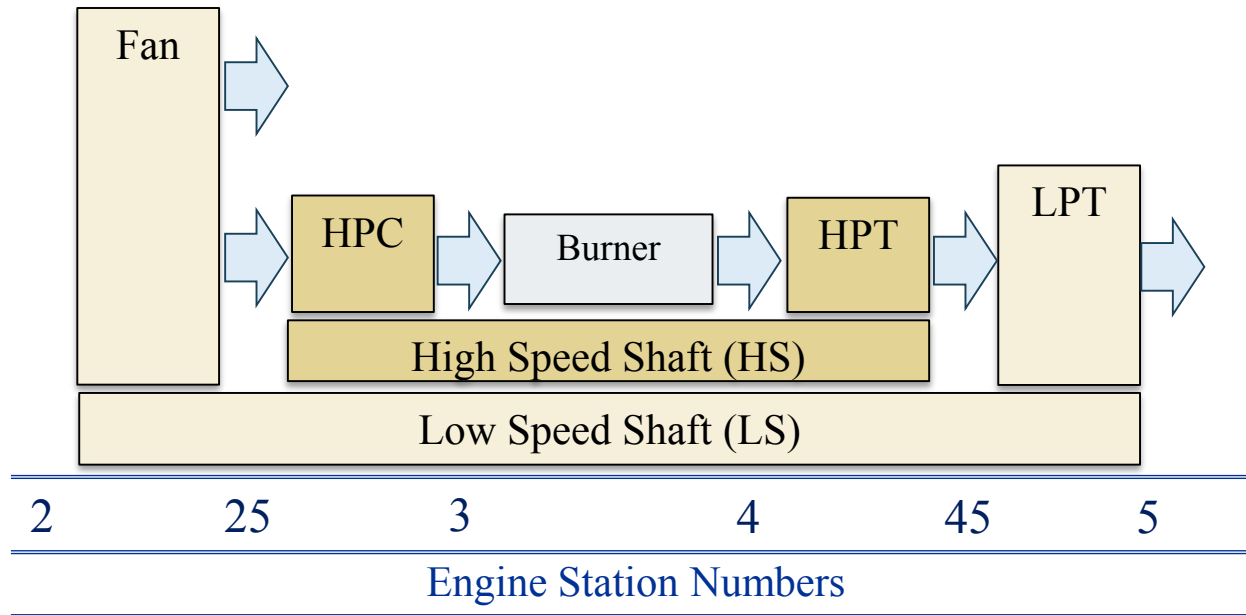
Engine Model

- Currently do not have access to physics based engine model
 - Model is required for Model-Based Engine Control (Estimated Parameters)
- Create engine model from engine data and available information
- Build model in T-MATS
 - Create component maps
 - Define operating line (region) on component maps
 - Scale maps to match data using the iDes tool of T-MATS to determine scalars and solve for missing information (geometry)





T-MATS Model



- No cooling bleeds
- No variable geometry
- *No customer bleed for cabin pressure (yet)*
- Geared turbofan
- Power extraction off HS to power aviation systems

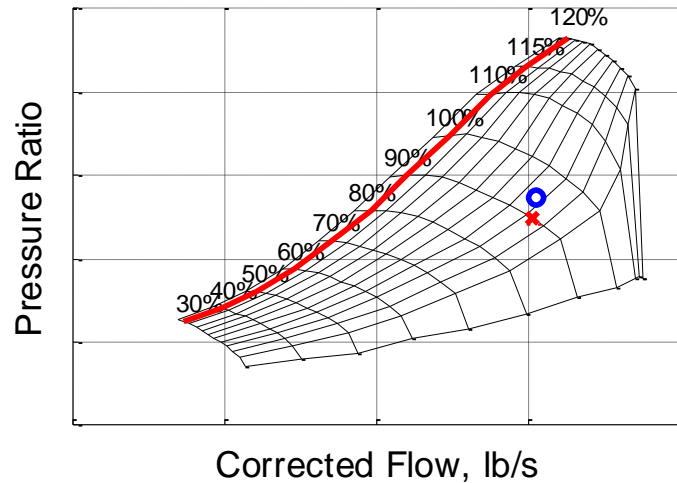
Combustor



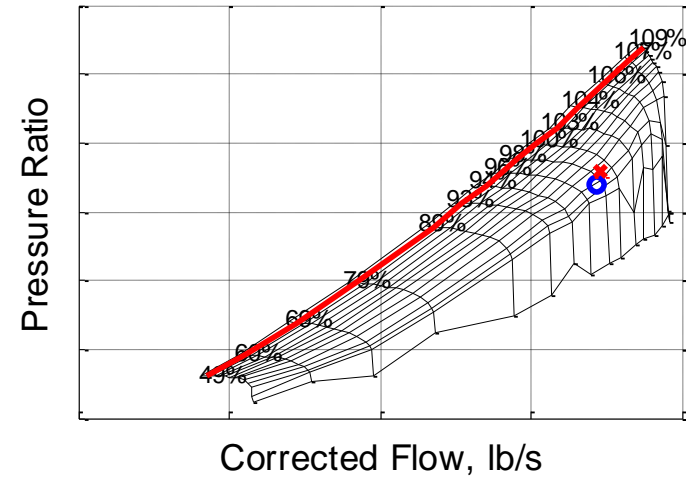
T-MATS Model

Compressor

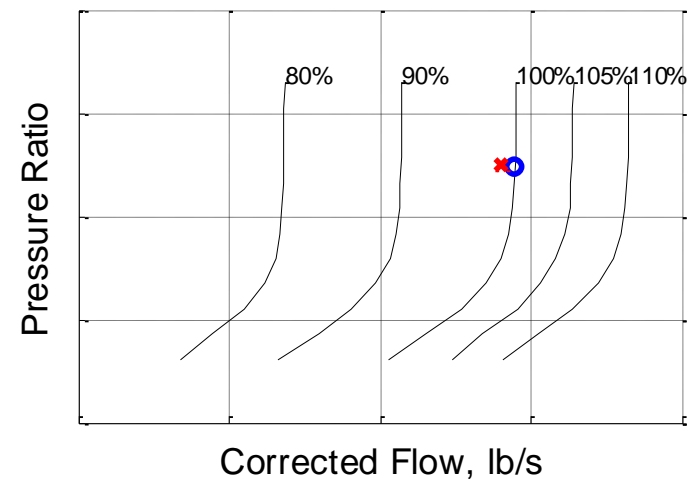
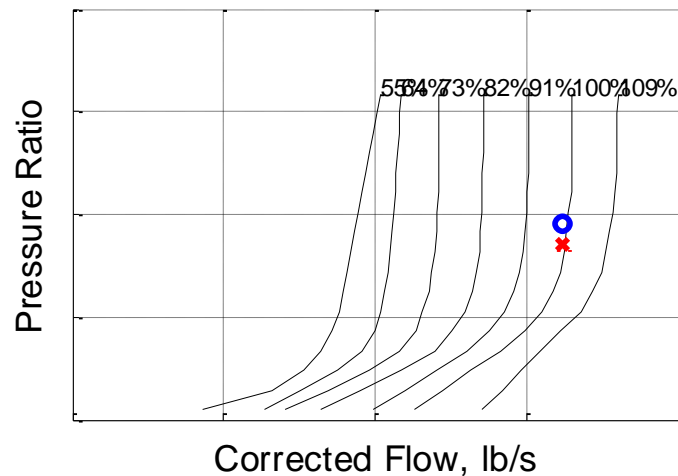
Low Speed



High Speed



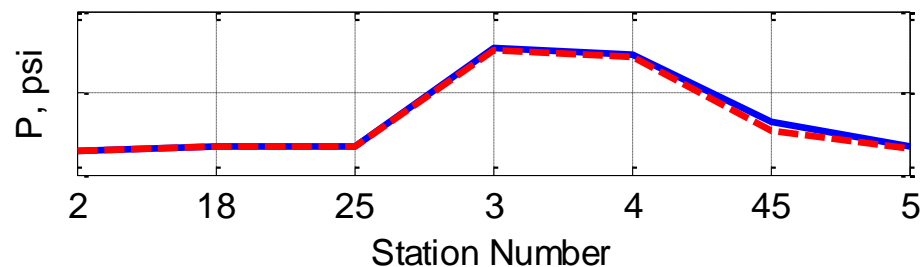
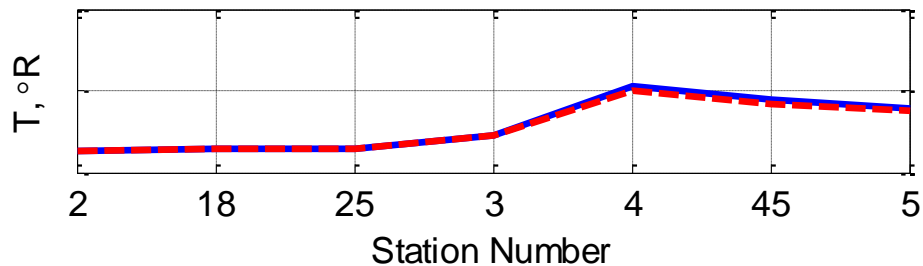
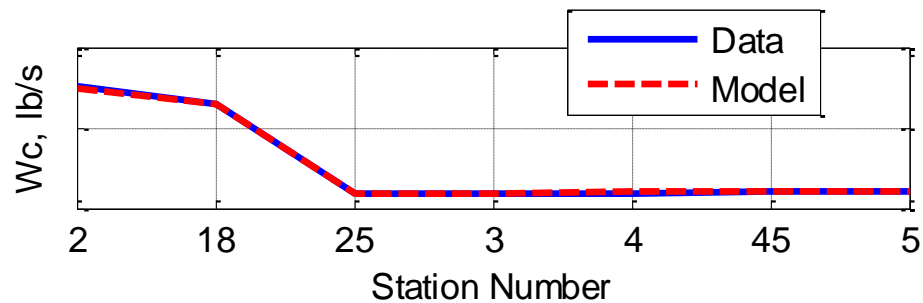
Turbines



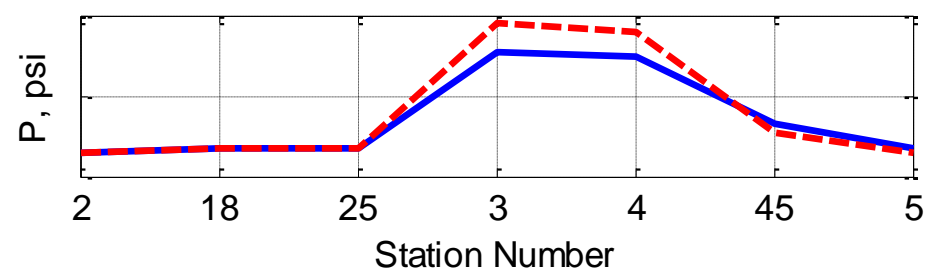
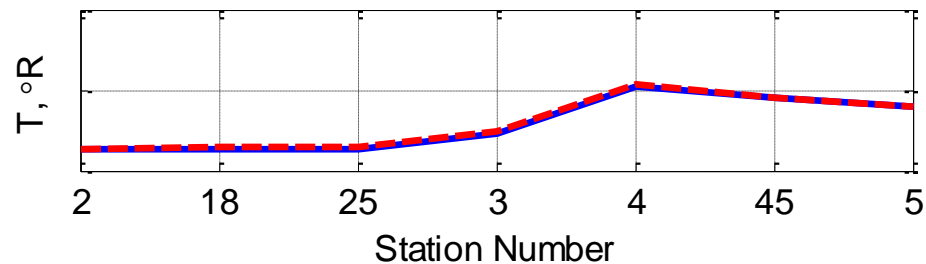
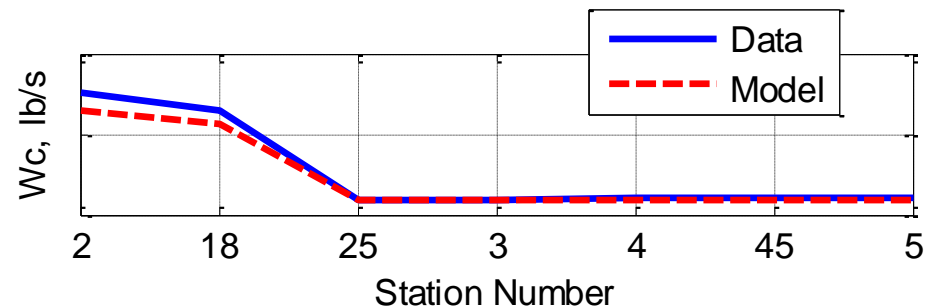


Model Comparison

Cruise Power Setting



Takeoff Power Setting





Summary/Future Work

- With limited Engine parameter information a reasonably accurate engine model for steady state was developed rapidly
- Continue to work on increasing accuracy of the model.
 - Add additional (T-MATS) components as required
 - Begin to use dynamic Jacobian solver and tune inertias
- Work on linear model development
 - T-MATS Linearization Function?
 - For MBEC/Kalman Filter
 - Offline - Create and store PWLM
 - Online - Use T-MATS model and linearize in real time
- Design Kalman Filters and MBEC Architecture
- Implement MBEC Architecture in Hardware platform for demonstration (*Maybe in T-MATS*)
- Final hardware implementation



Thank you!
Any Questions?